PUC Minas
A B A K Ó S
Instituto de Ciências Exatas e Informática

Uma Análise de Métodos de Autoaprendizado de Máquina

An Analysis of Machine Self-Learning Methods

Alessandra Faria Abreu[1]
Cristiane Neri Nobre (Orientadora)[2]

Resumo

Atualmente, as técnicas de aprendizado de máquina são utilizadas em quase todos os segmentos tecnológicos, como em aplicativos, campanhas de publicidade, análises de crédito e sistemas de recomendação. Na prática, as técnicas de aprendizado de máquina têm desempenhado um papel crucial ao permitir o aproveitamento da grande quantidade de dados produzida diariamente no meio digital. Em geral, o processo de desenvolvimento do aprendizado de máquina é um processo complexo, demorado e muito dependente dos conhecimentos e ações humanas. Para se obter um bom desempenho durante a execução dos modelos de aprendizado de máquina, os humanos estão fortemente envolvidos em todos os aspectos do processo de aprendizado. Inicialmente, é proposto neste trabalho um levantamento das principais ferramentas existentes na área de aprendizado de máquina automático, que visa automatizar diversas etapas do processo de aprendizagem de máquina. Num segundo momento, o *framework* de aprendizado de máquina automático *Auto sklearn*, escolhido neste trabalho como sendo um dos mais recomendados, foi aplicado a uma base de dados de presidiários americanos, com o objetivo de prever futuras infrações dentro do presídio. O uso do *framework Auto sklearn* resultou na melhora significativa no tempo de execução e de até dois pontos percentuais dos resultados, comparado com os trabalhos anteriores. Assim, espera-se que este trabalho possa servir não apenas como orientação criteriosa para iniciantes de aprendizado de máquina automático, mas também como inspiração para pesquisas futuras.

Palavras-chave: Machine Learning; AutoML; Aprendizado de Máquina automático.

[1]Graduação em Sistemas de Informação da PUC Minas, Brasil– alessandra.faria.abreu@gmail.com
[2]Programa de Pós Graduação em Informática da PUC Minas, Brasil– nobre@pucminas.br

Abstract

Currently, machine learning techniques are used in almost all technological segments, such as in applications, advertising campaigns, credit analysis and recommendation systems. In practice, machine learning techniques have played a crucial role in enabling you to take advantage of the large amount of data produced daily in the digital environment. In general, the machine learning development process is a complex, time-consuming process that is highly dependent on human knowledge and actions. To achieve good performance when executing machine learning models, humans are strongly involved in all aspects of the learning process. Initially, this work proposes a survey of the main tools in the area of automatic machine learning, which aims to automate various stages of the machine learning process. In a second step, the *framework* automatic machine learning *Auto sklearn*, chosen in this work as one of the most recommended, was applied to a database of American inmates, with the objective of predicting future infractions inside the prison. The use of the *Auto sklearn framework* resulted in a significant improvement in the execution time and up to two percentage points of the results, compared to previous works. Thus, it is hoped that this work can serve not only as a careful guide for beginners of automatic machine learning, but also as inspiration for future research.

Key words: Machine Learning; AutoML; Automatic Machine Learning.

1 INTRODUÇÃO

Para obter conhecimento sobre determinado assunto ,antes do uso da computação, era necessário realizar seguidas entrevistas com especialistas, visando entender quais variáveis e regras eram importantes para a execução de um processo estudado. No entanto, este método de obtenção de conhecimento é suscetível a erros, porque muitas vezes os especialistas podem não cooperar ou possuir o entendimento do processo como um todo, sendo necessária uma pessoa com alto entendimento do processo. Conforme os anos passaram, a complexidade dos processos e a quantidade de variáveis a serem tratadas aumentaram, de modo que as entrevistas podem se tornar inviáveis, bem como o uso de algoritmos computacionais em casos que possam ter muitas alternativas.

De acordo com (FACELI et al., 2011), os algoritmos computacionais necessários para realizar a análise e o levantamento das regras fundamentais em um processo de aprendizado devem ser algoritmos autônomos, capazes de aprender com experiências passadas.

A aprendizagem pode ser considerada fundamental para a identificação de um comportamento inteligente. Segundo (MITCHELL, 1997), aprendizado de máquina é "a capacidade de melhorar o desempenho na realização de alguma tarefa por meio da experiência". Monard e Baranauskas (2003) explica aprendizagem de máquina como a área de Inteligência Artificial que tem como objetivo o desenvolvimento de técnicas computacionais realizadas sobre um aprendizado, de modo que o sistema seja capaz de adquirir conhecimento automaticamente e tomar decisões baseadas em experiências de sucesso. Neste sentido, a aprendizagem de máquina trata-se de automatizar a tomada de decisões por meio de algoritmos capazes de treinar as máquinas.

Existem diversos algoritmos relacionados ao aprendizado de máquina. Definir os parâmetros que devem ser utilizados durante a execução de cada algoritmo, com o objetivo de controlar o processo de treinamento e garantir os melhores resultados torna-se inviável, pois cada problema se comporta de forma específica.

Os hiperparâmetros são explicados por (MONARD; BARANAUSKAS, 2003) como o conjunto de configurações de dados que controlaram o processo de treinamento do modelo de aprendizado de máquina. São ajustados após a análise do conjunto de resultados gerados pela execução de um treinamento na base de dados, visando encontrar a melhor combinação, em determinado algoritmo. Sem o ajuste automático dos hiperparâmetros, seria necessário que o modelo fosse modificado manualmente após cada execução de um treinamento. Existem no mercado diversas ferramentas que objetivam auxiliar o cientista de dados a ajustar melhor os hiperparâmetros do modelo a ser executado; um exemplo é o *software* de aprendizado de máquina *Weka* (HALL et al., 2009) que disponibiliza os algoritmos de *GridSearch* (LIASHCHYNSKYI; LIASHCHYNSKYI, 2019) , *MultiSearch* (KOTTHOFF et al., 2017) e *Auto-WEKA* (THORNTON et al., 2013).

Com o objetivo de diminuir os custos dispendiosos no processo de desenvolvimento, surgiu o conceito de automatizar todo o *pipeline* do aprendizado de máquina, ou seja, o desen-

volvimento de métodos de aprendizado de máquina automático (AutoML). Existem diversas definições para AutoML. De acordo com (ZÖLLER; HUBER,), o AutoML é projetado para reduzir a demanda por cientistas de dados e permitir que especialistas da área de negócio possam criar automaticamente modelos de aprendizado de máquina em aplicações que não demandem muito conhecimento estatístico. Yao et al. (2018) definem AutoML como uma combinação de algoritmos de aprendizados de máquina. Para Balaji e Allen (2018), o aprendizado de máquina automático automatiza a construção, o ajuste, a seleção e do processo de desenvolvimento do aprendizado de máquina.

Para este estudo, a base utilizada foi disponibilizada pelo Departamento de Justiça dos Estados Unidos, por meio de uma pesquisa contendo informações sobre pouco mais de 14 mil presidiários de prisões estaduais, abrangendo sentenças, antecedentes criminais, delitos, relacionamento com drogas, formação acadêmica, histórico familiar, dentre outros que, juntos, somam mais de 3 mil atributos.

Estudos realizados anteriormente por (NGO et al., 2015) e (PIANTINO, 2018) buscaram, utilizando técnicas de Redes Neurais Artificiais ou Árvores de Decisão, classificar os presidiários em usuários que quebraram ou não as regras estabelecidas pelas instituições. Nos estudos, os autores conseguiram um desempenho de 69,1% de precisão e afirmam que o problema principal da base é a quantidade de dados ausentes. O ajuste manual dos hiperparâmetros necessários para cada algoritmo foi inviável devido à quantidade e complexidade dos mesmos, bem como o uso de algoritmos computacionais e também, posteriormente, a automação destes algoritmos.

Este artigo possui, em seus objetivos, apresentar o levantamento dos *frameworks* de aprendizado de máquina automático existentes no mercado, explicando as características fundamentais de cada um, e realizar a aplicação do *framework Auto Sklearn* em uma base de dados prisionais americanos. Os resultados serão avaliados utilizando as melhores métricas de validação do aprendizado de máquina.

O restante deste trabalho está organizado em 6 seções. A Seção 2 apresenta o Referencial Teórico, onde são descritos os conceitos de aprendizado de máquina automático abordados no trabalho. Além disso, são descritos os principais *frameworks* de aprendizado de máquina automático, destacando o *framework Auto-Sklearn*, escolhido neste trabalho. Na Seção 3, é possível conhecer os principais estudos e resultados que tratam sobre aprendizado de máquina automático. A Seção 4 dedica-se aos materiais e métodos utilizados no trabalho, com a descrição da base de dados escolhida, os métodos que serão utilizados e as métricas de avaliação de qualidade que serão consideradas. A Seção 5 evidencia os resultados obtidos com as principais discussões. Por fim, na Seção 6 são realizadas as considerações finais e os trabalhos futuros.

2 REFERENCIAL TEÓRICO

São apresentados nesta seção os principais conceitos que envolvem a extração de conhecimento de algoritmos de aprendizado de máquina automático.

2.1 Hiperparâmetros

Os métodos de aprendizado de máquina são métodos matemáticos que utilizam diversos hiperparâmetros, consideráveis no ajuste dos modelos. Eles possuem propriedades importantes para o seu desempenho, tais como a rapidez e a complexidade no aprendizado.

Assim, ajustar os hiperparâmetros do modelo de aprendizado de máquina é fundamental para sua otimização. Em Pinto N. e Cox (2009), os autores demonstraram que é possível melhorar a classificação de imagens, melhorando o ajuste dos hiperparâmetros.

Cada um dos algoritmos de aprendizado de máquina pode ter vários hiperparâmetros. A Tabela 1 apresenta uma amostra da relação de alguns algoritmos de aprendizado de máquina e seus hiperparâmetros. Esta relação foi extraída de uma análise no *software* Weka (HALL et al., 2009), uma plataforma de aprendizado de máquina.

Tabela 1 – Descrição de alguns algoritmos e seus hiperparâmetros

Hiperparâmetro	Descrição
Árvore Aleatória	
Num Iterations	Número de interações a serem executadas no modelo
Batch size	Tamanho do conjunto desejado para previsão
Calc out bag	Define se o erro fora do conjunto será calculado
Num decimal places	Número de casas decimais a serem usadas na saída do modelo
Max depth	Número delimitador de profundidade da árvore
Num features	Número de atributos que devem ser considerados na execução
KMeans	
Distance function	Função de distância a ser executada no modelo
Don't replace missing values	Define se irá ou não repetir valores faltantes
Fast distance calc	Define se irá calcular as distâncias no modo rápido
Max interations	Define o número máximo de interações
Num clusters	Define qual o número de agrupamentos que devem ser gerados
Apriori	
Metric type	Define qual a métrica utilizada para ranquear os valores
Min metric	valor mínimo de *score* que a métrica escolhida irá assumir
Num rules	Número de regras/associações a serem encontradas
Lower bound min support	Suporte mínimo para o limite inferior
Rede Neural Artificial	
Batch size	Número predeterminado de instâncias a serem processadas
Learning rate	Taxa de aprendizado a ser considerada na atualização dos pesos
Normalize attributes	Define se irá normalizar os atributos automaticamente
Hidden Layers	Define as camadas serão ocultas no resultado

O ajuste de hiperparâmetros é realizado manualmente ou utilizando uma ferramenta

específica para isto. Isso faz com que o tempo e o custo para a execução de um modelo sejam maiores. Além disso, requer que o cientista de dados tenha o conhecimento específico do modelo que deve ajustar e de quais são as ferramentas necessárias para auxiliá-lo no ajuste. Atualmente, com a melhora da capacidade computacional, *clusters* e processadores são capazes de executar algoritmos para o ajuste de hiperparâmetros automaticamente.

2.2 Pré-processamento

A qualidade de dados é considerada uma das principais preocupações antes de realizar a aplicação de um método de aprendizado de máquina (BATISTA et al., 2003). Uma vez que a maioria dos algoritmos de aprendizado induz conhecimento estritamente a partir dos dados, a qualidade e quantidade do conhecimento extraído é diretamente determinada pela qualidade dos dados de entrada. Diversos aspectos podem influenciar no desempenho de um sistema de aprendizado devido à qualidade dos dados fornecidos. Utilizando bases de dados reais é comum enfrentar dois desafios que são apresentados por Batista et al. (2003), como a presença de valores desconhecidos que podem inferir sobre o resultado e a diferença entre o número de exemplos nas classes existentes.

O primeiro problema trata de como proceder quando as informações disponíveis na base de dados estão incompletas ou quando as fontes de informações se tornam indisponíveis. O tratamento dos dados ausentes deve ser realizado de forma planejada, pois do contrário, dados distorcidos serão introduzidos no processo de aprendizagem.

O segundo problema de aprender com dados desbalanceados é de significativa importância, pois pode ocorrer em diversas bases de dados. Sistemas de aprendizado de máquina assumem o uso de uma distribuição balanceada de dados; deste modo, classes desbalanceadas podem contribuir para um gargalo significativo no desempenho do método utilizado.

Com o objetivo de resolver estes e outros problemas relacionados ao processamento da base de dados, a etapa de pré-processamento é realizada antes da execução do aprendizado de máquina. Esta etapa tem a função de aprimorar a qualidade dos dados de entrada, para que os processos de aprendizado fiquem mais eficientes.

2.3 Aprendizado de máquina automático

O crescente uso de técnicas de aprendizado de máquina em diversas aplicações tornou necessário o uso de métodos que facilitassem sua execução. Em (AUTO-SKLEARN, 2019), é explicado que, para ser eficaz na prática, é preciso que o processo de aprendizado de máquina selecione automaticamente quais os algoritmos principais devem ser executados e quais hiperparâmetros devem ser ajustados. Em sua essência, o aprendizado de máquina automático deve fornecer ao desenvolvedor métodos que o auxiliem na otimização do modelo de ciência

de dados, no pré-processamento dos dados, na escolha dos hiperparâmetros e na avaliação dos resultados com métricas específicas (BALAJI; ALLEN, 2018).

Existem muitos *frameworks* para a execução do aprendizado de máquina automático, tais como MLBox (ROMBLAY, 2019), TPOT (AUTOML, 2019), H2O (H2O, 2017), Auto Keras (DATA LAB, 2019), Cloud AutoML (GOOGLE, 2018), Auto-Sklearn (FEURER et al., 2019) e muitos outros que auxiliam na etapa de pré-processamento e ajuste dos algoritmos e hiperparâmetros. Nas subseções seguintes, fornecida uma visão geral abrangente de diversas ferramentas e estruturas implementadas para automatizar o processo de seleção combinada de algoritmos e processo de otimização de hiperparâmetros.

2.3.1 MLBox

De acordo com a documentação oficial[1], *Machine Learning Box*, MLBox (ROMBLAY, 2019), é uma biblioteca da linguagem de programação *Python*, para o aprendizado de máquina automático, com três grandes fases que se segmentam em pré-processamento, otimização e aplicação.

Na etapa de pré-processamento são fornecidos os métodos relacionados à leitura e à limpeza da base de dados. Durante a leitura, a base de dados é processada e, com ela, são criados os conjuntos de teste e treino, que serão utilizados no momento da validação. Além disso, neste momento o MLBox identifica se deverá ser executado um algoritmo de regressão, classificação ou aprendizado por reforço. Na execução do método de limpeza dos dados são executadas as funções de pré-processamento e codificação das variáveis. Estas funções incluem:

- Codificação de valores categóricos;
- Eliminação de duplicatas;
- Conversão de datas em *timestamp*[2];
- Eliminação automática de *outliers*;

A etapa de otimização é a etapa de maior destaque do *framework* MLBox, pois nela é utilizada a biblioteca *hyperplot* que, além de promover resultados rápidos, oferece uma variedade de formas de otimização. Com o auxílio desta biblioteca, é criado um espaço de alta dimensão dos parâmetros a serem otimizados, o que permite ao MLBox escolher, de forma cruzada, a combinação dos parâmetros a serem ajustados que melhor promove a pontuação do modelo.

Na etapa de aplicação, diversos algoritmos de aprendizado de máquina são executados com os hiperparâmetros definidos e tratados nas etapas anteriores. Como resultado desta etapa, gera-se uma lista dos algoritmos executados com os parâmetros e métricas utilizados.

[1]Disponível em: https://mlbox.readthedocs.io
[2]Formato de data em milissegundos

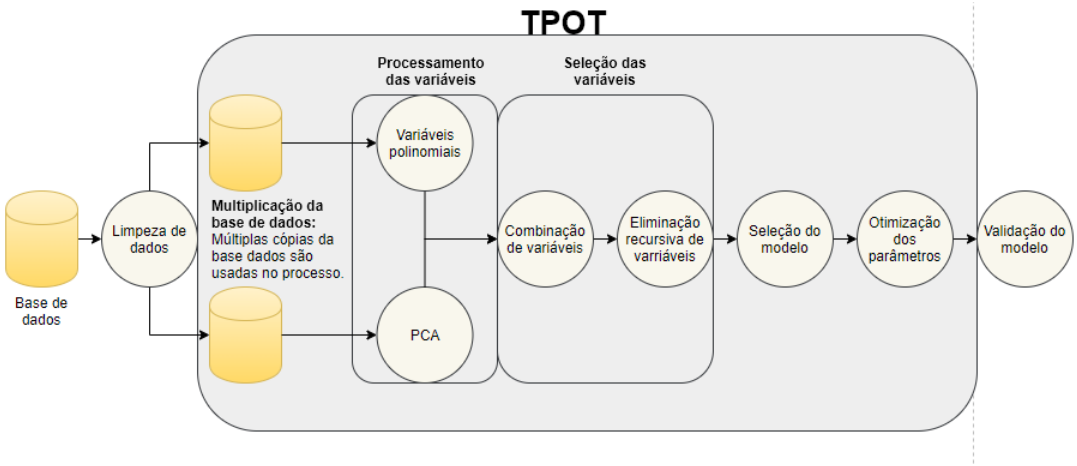
2.3.2 TPOT

TPOT, *Tree-Based Pipeline Optimization Tool*, é uma ferramenta que otimiza o *pipeline* de aprendizagem de máquina de forma genética e baseada em árvore. Ela automatiza a etapa mais morosa do processo de aprendizagem de máquina, explorando de modo inteligente milhares de possibilidades de execução dos algoritmos de aprendizado de máquina até encontrar o melhor resultado (OLSON et al., 2016).

Com o TPOT é possível executar algumas funções relacionadas ao tratamento das variáveis e seleção do modelo. Para tratamento das varáveis podem ser utilizadas as funções de seleção e pré-processamento. Na seleção do modelo podem ser utilizadas as funções de seleção e otimização do modelo. Como limitação, o TPOT não permite processar dados de linguagem natural e nem sequências categóricas. Estas devem ser transformadas em números inteiros antes do processamento.

A Figura 1 apresenta o processo de aprendizado de máquina supervisionado e com ela é possível observar, em destaque, as etapas do *pipeline* que o TPOT pode automatizar.

Figura 1 – Aprendizado de máquina com TPOT



Fonte: Elaborado pelo autor, com dados extraídos de (OLSON et al., 2016)

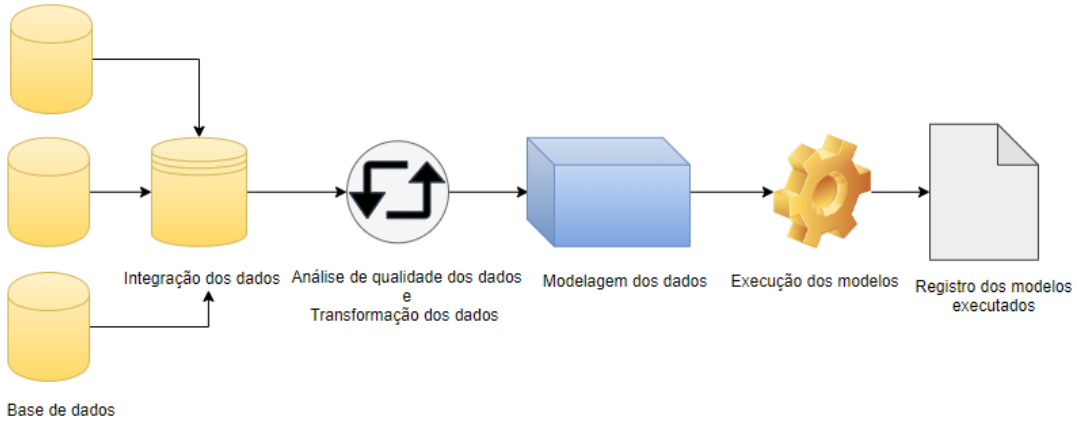
2.3.3 H2O

O H2O é uma plataforma de código aberto, de aprendizado de máquina automático com suporte para as linguagens de programação R e *Python*. Para iniciar a escolha dos modelos a serem testados, ele utiliza o algoritmo *Randon Search*, apresentando em (BERGSTRA; BENGIO, 2012) como um algoritmo que realiza pesquisas de modelos de forma aleatória e consegue encontrar melhores modelos, pesquisando efetivamente em espaços de configurações maiores.

Dentre os modelos que podem ser escolhidos pelo H2O, destacam-se os de regressão linear e logística, algoritmos baseados em árvore e alguns algoritmos que utilizam o método gradiente como otimização. Os resultados gerados exibem, em sequência, o modelo, a acurácia

obtida e a perda. A Figura 2 apresenta, em visão macro, o *pipeline* executado pelo *framework* H2O.

Figura 2 – Aprendizado de máquina com H2O



Fonte: Elaborado pelo autor, com dados extraídos de (H2O, 2017)

2.3.4 Auto Keras

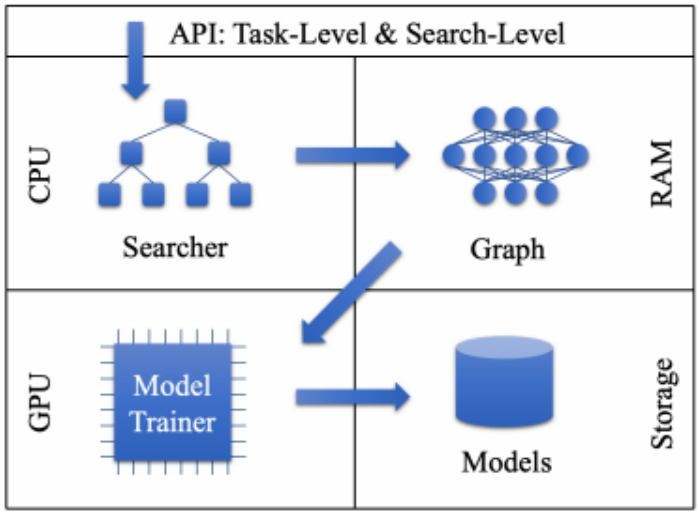
Auto Keras é uma biblioteca de código aberto que emprega o NAS (*Neural Architecture Search*) com otimização bayesiana que, diferente dos outros *frameworks*, concentra-se nas tarefas de aprendizado profundo. Segundo Jin et al. (2019), o Auto Keras é um sistema de código aberto que foi cuidadosamente planejado para oferecer uma interface para pessoas não especialistas.

A Figura 3 apresenta a visão geral do sistema Auto Keras. Com a figura é possível identificar os procedimentos executados pelo *framework* (JIN et al., 2019):

1. O usuário chama a API;
2. O *Searcher* gera arquiteturas neurais na CPU;
3. *Graph* cria redes neurais reais com parâmetros na RAM das arquiteturas neurais;
4. A rede neural é copiada para a GPU para o treinamento.
5. Redes neurais treinadas são salvas em dispositivos de armazenamento.

Durante seu processamento, o Auto Keras busca encontrar automaticamente arquiteturas de modelos de aprendizado profundo, além de otimizar seus respectivos hiperparâmetros. O AutoKeras começa com um modelo simples e continua a construir modelos até o limite de tempo especificado. O melhor modelo é escolhido com base na pontuação de perda e precisão. A precisão do modelo depende do conjunto de dados e do limite de tempo disponibilizados para o treinamento.

Figura 3 – Visão geral do sistema Auto-Keras.



Extraído de (JIN et al., 2019)

2.3.5 Cloud AutoML

O Cloud AutoML é um conjunto de produtos dispostos em uma plataforma *online*, desenvolvido pela Google, especializado em aprendizado de máquina. Ele permite que qualquer desenvolvedor, mesmo iniciante, possa criar modelos de aprendizado de máquina de alta qualidade. A Figura 4 apresenta os produtos de aprendizado de máquina disponibilizados no Cloud AutoML. Os produtos são divididos em recursos de Visão, Idioma e Dados Estruturados.

Figura 4 – Produtos Cloud AutoML

Fonte: Elaborado pelo autor, com dados extraídos de (GOOGLE, 2018)

A plataforma não possui código aberto e seu custo é calculado por demanda.

2.3.6 Auto-Sklearn

O *framework* Auto-sklearn é o resultado de uma pesquisa realizada na Universidade de Freiburg destinado ao pré processamento dos dados, à seleção de algoritmos e o ajuste de hiperparâmetros. Foi desenvolvido sobre a biblioteca *Scikit-learn*, uma das mais conhecidas e utilizadas para o aprendizado de máquina. Ele é composto por um conjunto de ferramentas para o aprendizado de máquina automático.

Diferente dos demais *frameworks*, o Auto-sklearn possui dois métodos desenvolvidos por (JIN et al., 2019) para aumentar a robustez e a eficiência do aprendizado. Primeiro, é a inclusão de um meta-aprendizado para iniciar de forma rápida o passo da otimização bayesiana, o que resulta em um aumento considerável de eficiência. Em segundo lugar, o *framework* inclui uma etapa para a construção automatizada do modelo, permitindo utilizar todos os classificadores encontrados durante a otimização bayesiana.

A área de meta-aprendizado imita a estratégia de extrair o conhecimento de tarefas anteriores. No Auto-sklearn essa área é utilizada para selecionar parâmetros da estrutura de aprendizado de máquina que possam ter um bom desempenho em um novo conjunto de dados. É uma abordagem complementar à otimização bayesiana.

A vantagem do meta-aprendizado está no fato de ele sugerir de forma rápida algumas instanciações que, quando usadas, alcançam um bom desempenho; contudo, o meta-aprendizado é incapaz de fornecer as informações de forma detalhada. A otimização bayesina é considerada lenta, quando usada em situações em que a quantidade de hiperparâmetros é muito grande. Por outro lado, ela permite ajustar o desempenho ao longo do tempo. Aproveitando-se dessa complementariedade, o *framework* Auto-sklearn usa K configurações do meta-aprendizado para direcionar a otimização bayesiana.

Os quinze algoritmos de classificação utilizados pelo Auto-sklearn que foram estudados pelo Jin et al. (2019) são listados na Figura 5(a). Eles podem se enquadrar em categorias diferentes, como linear (2 algoritmos), máquinas de vetores de suporte (2), análise discriminante (2), vizinhos mais próximos (1), Bayes (3), árvores de decisão (1) e conjuntos (4).

Os métodos de pré-processamento de conjuntos de dados listados na Figura 5(b) incluem: redimensionamento, que é diminuição da quantidade de atributos por métodos estatísticos; imputação de valores ausentes; codificação *one-hot*, que é a transformação de valores categóricos em valores inteiros; e balanceamento das classes. Além disso, os métodos podem ser classificados e divididos em: seleção de recurso (2), aproximação de *kernel* (2), decomposição de matriz (3), *embeddings* (1), *feature clustering* (1), expansão de recurso polinomial (1) e métodos que usam um classificador para seleção de variáveis (2).

A Figura 5 mostra o número de hiperparâmetros para cada classificador possível (a) e o método de pré-processamento de recursos (b), que estão diferenciados entre hiperparâmetros categóricos (cat) com valores discretos e hiperparâmetros numéricos contínuos (cond). Os números entre parênteses são os hiperparâmetros condicionais, que são relevantes apenas quando outro parâmetro tem um determinado valor.

Figura 5 – Algoritmos (a) e métodos (b) utilizados pelo Auto-Sklearn

name	#λ	cat (cond)	cont (cond)
AdaBoost (AB)	4	1 (-)	3 (-)
Bernoulli naïve Bayes	2	1 (-)	1 (-)
decision tree (DT)	4	1 (-)	3 (-)
extreml. rand. trees	5	2 (-)	3 (-)
Gaussian naïve Bayes	-	-	-
gradient boosting (GB)	6	-	6 (-)
kNN	3	2 (-)	1 (-)
LDA	4	1 (-)	3 (1)
linear SVM	4	2 (-)	2 (-)
kernel SVM	7	2 (-)	5 (2)
multinomial naïve Bayes	2	1 (-)	1 (-)
passive aggressive	3	1 (-)	2 (-)
QDA	2	-	2 (-)
random forest (RF)	5	2 (-)	3 (-)
Linear Class. (SGD)	10	4 (-)	6 (3)

(a) classification algorithms

name	#λ	cat (cond)	cont (cond)
extreml. rand. trees prepr.	5	2 (-)	3 (-)
fast ICA	4	3 (-)	1 (1)
feature agglomeration	4	3 ()	1 (-)
kernel PCA	5	1 (-)	4 (3)
rand. kitchen sinks	2	-	2 (-)
linear SVM prepr.	3	1 (-)	2 (-)
no preprocessing	-	-	-
nystroem sampler	5	1 (-)	4 (3)
PCA	2	1 (-)	1 (-)
polynomial	3	2 (-)	1 (-)
random trees embed.	4	-	4 (-)
select percentile	2	1 (-)	1 (-)
select rates	3	2 (-)	1 (-)
one-hot encoding	2	1 (-)	1 (1)
imputation	1	1 (-)	-
balancing	1	1 (-)	-
rescaling	1	1 (-)	-

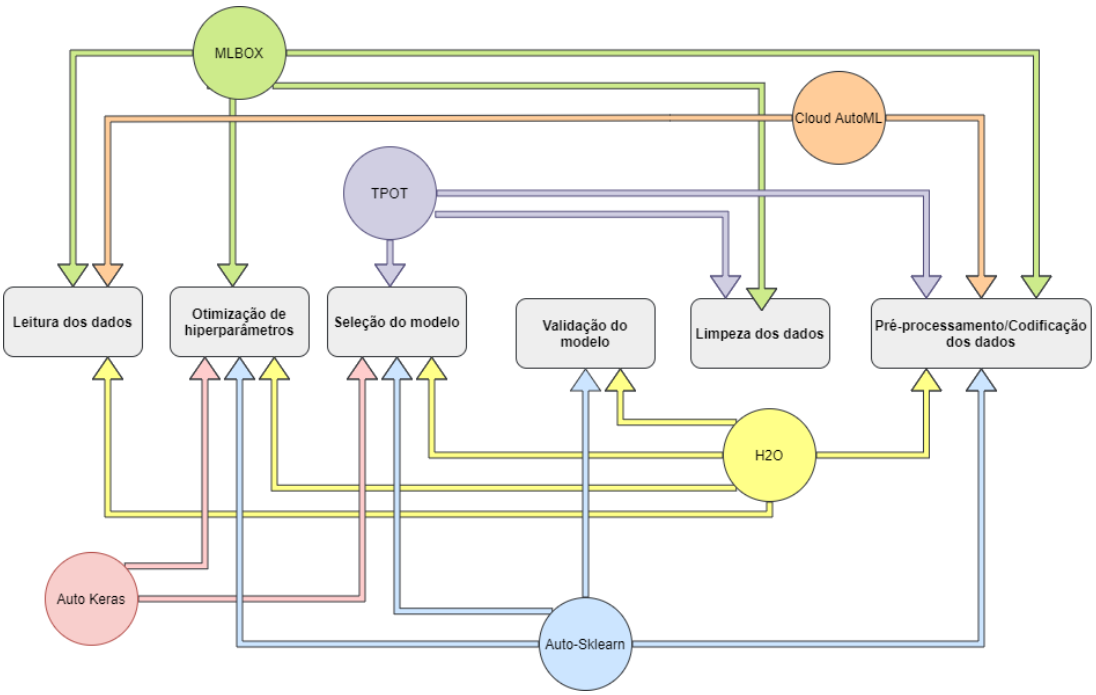
(b) preprocessing methods

Fonte: Extraído de (JIN et al., 2019)

2.3.7 Análise dos pontos de convergência dos frameworks

Os princípios do aprendizado de máquina automáticos perpassam pelas etapas de leitura dos dados, limpeza dos dados, codificação/pré-processamento dos dados, seleção de modelo, otimização de hiperparâmetros e validação do modelo. A Figura 6 apresenta as etapas que os *frameworks* de autoaprendizado de máquina, levantados neste trabalho, são capazes de realizar.

Figura 6 – Comparação de *frameworks* de autoaprendizado de máquina



Fonte: Elaborado pelo autor, com dados extraídos com a execução deste trabalho

Os *frameworks* H2O, Auto-Sklearn e MLBOX são, nesta ordem, os *frameworks* que possuem maior número de etapas automáticas; por outro lado, o Auto Keras e o Cloud AutoML

são os *frameworks* que menos possuem etapas automáticas. Quando entendido sob o ponto de vista de desempenho e de aplicabilidade, o Auto-Sklearn se destaca, pois além de ter a natureza do código aberta, compreende etapas julgadas como necessárias para a avaliação do aprendizado automático e tem a possibilidade de ajustar uma quantidade de hiperparâmetros maior que a do H2O. Deste modo, é justificável a escolha do Auto-Sklearn como *framework* a ser utilizado neste trabalho.

3 TRABALHOS RELACIONADOS

São apresentados nesta seção os trabalhos que, com a utilização do aprendizado de máquina automático, otimizaram o processo de aprendizagem de máquina considerado custoso e complexo para o desenvolvimento humano.

O trabalho Feurer et al. (2019) teve grande contribuição acadêmica, pois nele é apresentado o *framework Auto-Sklearn* , capaz de utilizar otimização bayesiana, para ajustar os hiperparâmetros, e meta-aprendizado, para agrupar dados semelhantes. Neste trabalho obtém-se, como resultado, um sistema com desempenho superior ao estado da arte em aprendizado de máquina automático.

Em (SHAWI et al., 2019) é realizado um amplo estudo acerca dos *frameworks* de aprendizado de máquina existentes. Neste trabalho, é apresentado como cada *framework* funciona, considerando o seu algoritmo central, a linguagem em que foi codificado e principalmente o seu custo financeiro. Shawi et al. (2019) preocupam-se em apresentar uma pesquisa abrangente nos esforços do aprendizado de máquina automático para possibilitar menores custos financeiros aos desenvolvedores.

Feurer et al. (2019) realizaram um estudo em que o aprendizado de máquina automático deveria produzir, sem nenhuma contribuição humana, um conjunto de testes válidos para um novo conjunto de dados dentro de um orçamento computacional fixo (tempo de CPU e uso de memória). Para chegar ao objetivo do estudo, foi utilizado o *framework* Auto-Sklearn, em que seu resultado foi melhor que o estado da arte em aprendizado de máquina automático, proporcionando mais eficiência e robustez.

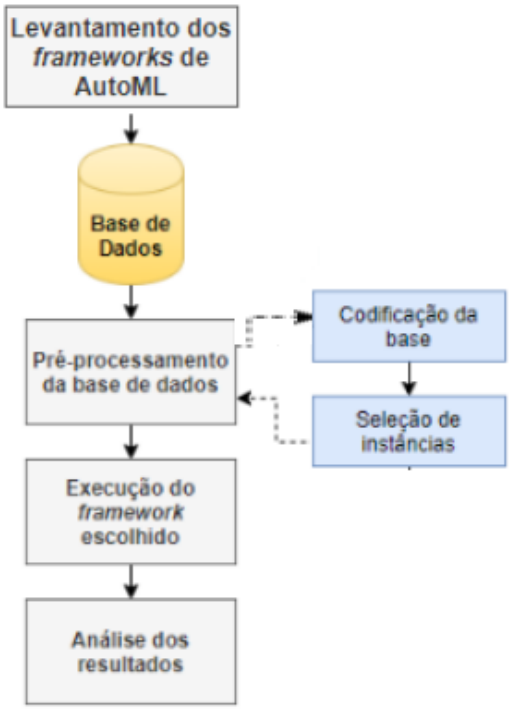
É apresentado por Gijsbers et al. (2019) o problema de comparar diferentes sistemas de aprendizado de máquina, pois geralmente é feita de maneira incorreta onde, geralmente, há dados com escopo limitado e desatualizado, e o tempo de execução não é proporcionalmente ajustado. Em seu trabalho foi introduzida uma estrutura de *benchmark*[1] aberta, contínua e extensível, que segue as melhores práticas para evitar erros comuns. A estrutura proposta possui código aberto, usa conjuntos de dados públicos e possui um *site* com resultados atualizados. Foi utilizada uma estrutura para realizar a comparação completa dos sistemas de aprendizado de máquina automáticos avaliados em 39 conjuntos de dados.

[1]Processo de busca das melhores práticas de gestão.

4 MATERIAIS E MÉTODOS

Nesta seção, é apresentado a metodologia adotada nesta pesquisa. A Figura 7 descreve todas as fases da metodologia e as seções a seguir explicam cada uma delas.

Figura 7 – Metodologia utilizada neste trabalho



Fonte: Elaborado pelo autor do trabalho

4.1 Coleta dos dados

A base de dados utilizada neste trabalho é resultado de uma Pesquisa de Presos em estabelecimentos prisionais estaduais e federais dos Estados Unidos - SIS-FCF (JUSTICE, 2004). Esta base foi desenvolvida por meio de pesquisas realizadas com detentos, dentro de diversos estados federais e prisões estaduais, no intervalo de tempo de outubro de 2003 a maio de 2004. Com esta base de dados, pretende-se prever se determinado presidiário cometerá alguma infração dentro do instituto prisional.

As informações sobre os presidiários, coletadas durante a pesquisa, abordam o crime, a sentença atual, os antecedentes criminais, o histórico familiar, as características pessoais, o histórico de uso de drogas (legal ou não), a posse e uso de armas, os dados sobre a participação em programas de reabilitação e as atividades realizadas dentro da instituição correcional (trabalho, estudos, religião, dentre outros).

Com o objetivo de melhor selecionar os presos que irão compor a amostragem de dados foram realizadas duas etapas com base estatística, levando em consideração gênero, localidade

e nível de segurança, como alguns dos grupos relevantes.

Na primeira etapa, de um conjunto de 1435 instituições estatais masculinas com 1.115.853 presos e 366 instituições estatais femininas com 77.404 presas, foram selecionadas 231 instituições masculinas e 70 instituições femininas. Na segunda etapa, das instituições conhecidas e levando em conta os fundamentos estatísticos, 13.098 homens e 3.054 mulheres presos foram selecionados. Desse grupo, 11.569 homens e 2.930 mulheres foram entrevistados.

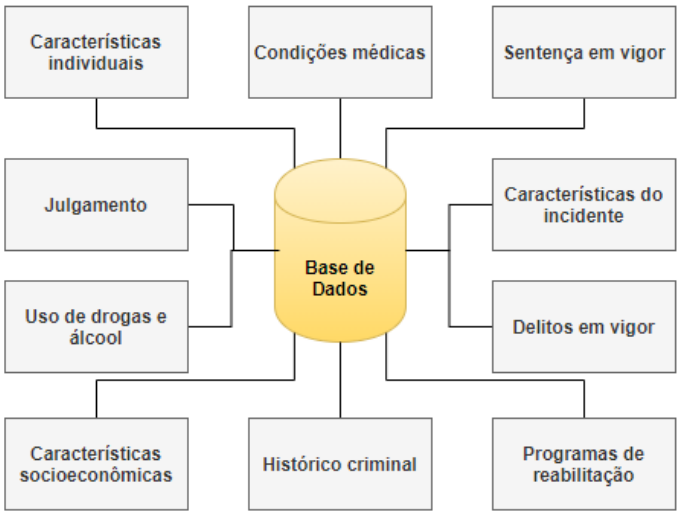
Este mesmo processo foi empregado para as instituições federais. Dentre 131 instituições federais masculinas, com 121.000 presos e 17 instituições federais femininas, que possuíam 9.419 presos, foram escolhidas 32 instituições masculinas e 8 femininas. Destas, foram selecionados 3.244 homens e 1.009 mulheres, dos quais 2.728 homens e 958 mulheres foram entrevistados.

É importante ressaltar que todos os presidiários participaram voluntariamente e que os dados coletados foram anônimos. Não houve nenhuma ligação entre os indivíduos e as informações coletadas. No total, 18.185 presos foram entrevistados.

4.2 Descrição da base de dados

Dentre os presos entrevistados, 14.499 foram consolidados como instâncias na base de dados final e o número de atributos foi de 2.980. Os atributos da base de dados são divididos em 2.046 valores numéricos discretos e 940 valores numéricos contínuos, organizados em 10 diferentes categorias, conforme ilustrado na Figura 8.

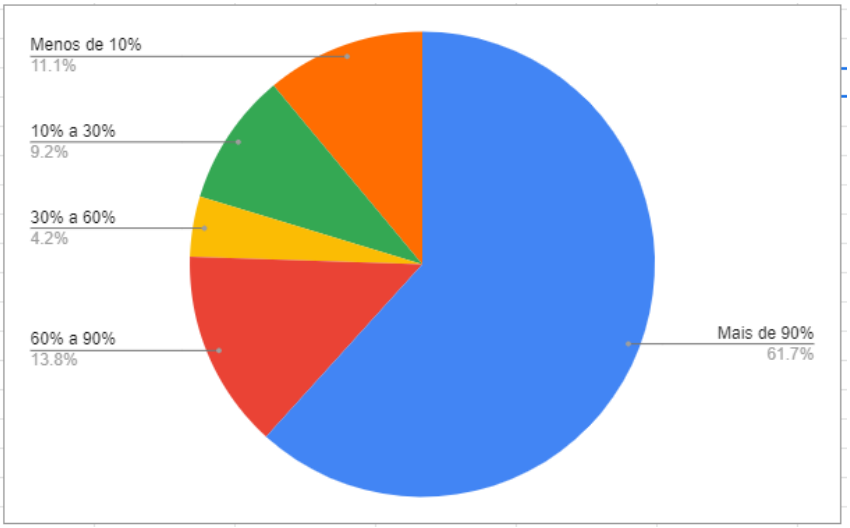
Figura 8 – Distribuição dos atributos da base de dados, distribuído por categorias



Fonte: Criado pelo autor com dados extraídos de SIS-FCF (JUSTICE, 2004)

Na base de dados original, 376 atributos estão totalmente ausentes, 1876 estão com mais de 90% de ausência e 2.299 com mais de 50% de ausência. Em contrapartida, 138 atributos estão em sua totalidade presentes, 331 estão com mais de 90% dos valores presentes e 687 estão com mais de 50% dos valores presentes; estes valores estão ilustrados na Figura 9.

Figura 9 – Porcentagem de ausência por atributos



Fonte: Elaborado pelo autor, com dados extraídos de SIS-FCF (JUSTICE, 2004)

Além da base de dados original, esse trabalho contempla uma versão filtrada. Uma base de dados composta pelas mesmas instâncias do original, mas com atributos reduzidos, especificamente 249, selecionados por especialistas de diversas áreas relacionadas à criminologia. Analisando a versão filtrada com os atributos selecionados e com base nos estudos realizados por Piantino (2018), foram identificados atributos que trazem problemas para a classificação final, atributos que eram englobados no atributo de classificação e geravam regras espúrias e falsos resultados na classificação.

De todos os atributos em ambas as versões da base de dados, os 15 atributos da Tabela 2 identificam quais violações o preso cometeu. Todos esses atributos possuem valores binários sim/não, que determinam se o preso foi considerado culpado da violação em questão. Contudo, o objetivo é predizer se um preso quebrou ou não uma regra e não qual o tipo de regra que este preso quebrou; o atributo de classificação da base de dados é o atributo "*Written Up Or Found Guilty Of Breaking Any Rules*" (v2517).

Outros atributos foram identificados como geradores de regras espúrias (PIANTINO, 2018). Desta forma, eles também foram retirados (Tabela 3). O atributo v2540 foi removido, pois também engloba os 15 atributos mencionados anteriormente. O atributo v2549 identifica quais das 15 violações foi a mais recente. O atributo v2550 informa se uma ação disciplinária foi tomada para a violação e os atributos de numeração v2551 a v2562 informam qual ação disciplinária foi tomada. Todos foram removidos devido à sua conexão com o atributo v2517 para evitar regras espúrias. Foi decidido retirar os atributos com mais de 50% ausência, 106 de 215 atributos.

Através de análises realizadas na base foram removidos atributos de controle da prisão e atributos que questionavam sobre o tratamento de saúde que o preso deveria ter recebido ao ser admitido (Tabela 4). Desta forma, a base de dados resultou em 109 atributos finais.

Tabela 2 – Identificação do Tipo de Violação

Atributo	Nome do Atributo
v2518	Guilty Of Drug Violation
v2520	Guilty Of Alcohol Violation
v2522	Guilty Of Possession Of A Weapon
v2524	Guilty Of Possession Of Stolen Property
v2526	Guilty Of Possession Of Other Unauthorized,
v2528	Guilty Of Verbal Assault On Staff Member
v2530	Guilty Of Physical Assault On Staff Member
v2532	Guilty Of Verbal Assault On Another Inmate
v2534	Guilty Of Physical Assault On Another Inmate
v2536	Guilty Of Escape Or Attempted Escape
v2538	Guilty Of Being Out Of Place
v2540	Guilty Of Disobeying Orders
v2542	Guilty Of Any Other Major Violation
v2544	Guilty Of Any Minor Violation
v2546	Guilty Of Any Other Violation

Tabela 3 – Geradores de Regras Espúrias

Atributo	Nome do Atributo
v2540	Guilty Of Rules Violation
v2549	Which Violations More Recent
v2550	Any Disciplinary Action Take Place
v2551-62	Disciplinary Actionn

Tabela 4 – Atributos de Controle

Atributo	Nome do Atributo
v2269	At Admission Check For Sick Injury Or Intoxication
v2270	At Admission Questioned About Health
v2271	At Admission Questioned About Suicide
v2274	Since Admission Test TB (Tuberculosis)

Analisando as instâncias, foram encontradas 258 das 14.499 instâncias, ou 1,7%, com o atributo de classificação ausente. Em algumas destas instâncias (11 de 258), foi possível determinar a classificação, pois possuía classificação positiva em algum dos atributos da Tabela 2 ou da Tabela 3. As instâncias que não haviam classificação nos atributos foram removidas. Com isso, a base de dados final contém 14.252 instâncias.

4.3 Processamento da base de dados

Na base de dados a maioria dos atributos eram compostos por dois valores nominais de categorias como sim/não ou masculino/feminino. Estes atributos foram codificados como valores numéricos dicotômicos 0/1.

Alguns atributos eram do tipo categórico e poderiam ter mais de um valor como, por exemplo, 'Tipo de Ofensa', que possui mais de 4 opções de resposta. Esses atributos foram binarizados, ou seja, tiveram suas respostas divididas em atributos sim/não individuais e posteriormente, também foram codificados como valores numéricos dicotômicos 0/1.

Os atributos numéricos, como idade, foram todos normalizados. Os novos valores foram gerados usando uma função de normalização dada pela Equação 1, onde o valor normalizado (*Vnovo*) é determinado por operações entre os valores de mínimo e de máximo (*min* e *max*) para aquele atributo, juntamente com o valor sem normalização (*Vatual*). Esse processo consiste em organizar todos os valores do atributo em uma nova escala de valores, entre valores de 0 e 1, para todos os atributos, impedindo, dessa forma, que um valor tenha influência sobre o outro (IKEDA, 2011). Vale ressaltar que *outliers* destes atributos foram configurados como ausentes e, em seguida, imputados utilizando a equação 1.

$$v_{novo} = min + \frac{v_{atual} - min}{max - min} * (max - min) \quad (1)$$

Assim, a base de dados resultante é composta apenas de valores numéricos entre 0 e 1. Após todos os processamentos, tanto dos atributos quanto de instâncias, temos uma base final com 144 atributos e 14.252 instâncias.

4.4 Descrição dos Métodos

O desenvolvimento deste trabalho pode ser dividido em duas grandes etapas: primeiramente, a execução do *framework* de aprendizado de máquina automático *autosklearn*. Após isto, são utilizadas as métricas de avaliação da qualidade dos modelos obtidos.

As subseções 4.4.1. e 4.4.2. apresentam os passos e desafios encontrados neste desenvolvimento.

4.4.1 Execução framework Auto sklearn para o aprendizado de máquina automático

Para o treinamento dos modelos de aprendizado de máquina, primeiramente 30% das amostras da base de dados foram separadas para realizar a avaliação da qualidade dos modelos obtidos pelo Autoaprendizado. Essa divisão foi realizada com *randon state*[1] igual a 42, como aconselha a documentação oficial *Sklearn* (PEDREGOSA et al., 2011).

Com objetivo de executar o aprendizado de máquina automático utilizando o *framework Autosklearn*, após a divisão da base de dados em treino e teste, a biblioteca *autosklearn classification* foi implementada no código do trabalho. Após a importação da biblioteca foram realizadas algumas parametrizações para as execuções dos modelos de aprendizado de máquina. As configurações parametrizadas no *autosklearn classification* foram o tempo de execução to-

[1]Gerador interno de números aleatórios, que decidirá a divisão dos dados em índices de treino e testes

tal do treinamento, o tempo máximo de execução para cada modelo a ser testado e ,em alguns casos, o modelo que a ser ajustado. A Tabela 5 mostra os parâmetros utilizados para os testes do *framework autosklearn* usando o tempo máximo de execução permitido no ambiente de desenvolvimento.

Tabela 5 – Parâmetros *autosklearn classification*

Teste	Tempo max de execução	Tempo max por modelo	Modelo
1	Seis horas	Uma hora	SVM
2	Seis horas	Uma hora	Randômico
3	Seis horas	Uma hora	Gradient boosting
4	Dez horas	Uma hora	Randômico
5	Seis horas	Uma hora	AdaBoost

No Teste 1, o modelo foi previamente parametrizado com o objetivo de validar as conclusões realizadas no trabalho (PIANTINO, 2018), que realiza a classificação de possíveis infrações na mesma base de dados, utilizando diversos práticas de imputação de dados, em sua conclusão Piantino (2018) o método de classificação SVM apresentou os melhores resultados. Deste modo, no Teste 1, o *framework* foi utilizado para, somente, ajustar os hiperparâmetros possíveis para o modelo SVM. No Teste 2 e no Teste 4, o *framework* foi utilizado para realizar o pré-processamento dos dados, escolher os melhores modelos e ajustar os hiperparâmetros possíveis; para obter resultados diferentes, o Teste 4 foi realizado com um tempo de execução maior e os melhores modelos encontrados foram *Gradient boosting* e *Adaboost* em sequencia. Com o resultado do melhor modelo definido pelo Teste 2, o Teste 3 foi utilizado para, somente, realizar o pré-processamento e ajustar os hiperparâmetros possíveis do modelo encontrado no Teste 2, com o objetivo de alcançar melhores resultados. No Teste 5, foi parametrizado o método de classificação encontrado pelo Teste 4.

4.4.2 Métricas de avaliação de qualidade

Para todos os métodos de classificação executados foram utilizadas as métricas precisão, sensibilidade e *f-measure* para avaliar o desempenho dos resultados obtidos. Suas respectivas fórmulas foram descritas abaixo, onde VP (Verdadeiro Positivo): quantidade de detentos corretamente classificados em uma determinada classe; FN (Falso Negativo): quantidade de detentos da classe analisada, erroneamente classificados; FP (Falso Positivo): detentos que não são da classe considerada, mas que foram classificados nesta classe.

1) Precisão: Mede a proporção de instâncias que foram identificadas corretamente como pertencentes à classe em questão, calculada pela Equação 2.

$$Precisao = \frac{VP}{VP + FP} \tag{2}$$

2) Sensibilidade (*Recall*): Mede a porcentagem de instâncias da classe que realmente foram classificadas como sendo da classe, calculada pela Equação 3.

$$Recall = \frac{VP}{VP + FN} \tag{3}$$

3) *F-Measure*: Média harmônica entre a Precisão e *Recall*, calculada pela Equação 4.

$$F - Measure = 2 * \frac{Precisao * Recall}{Precisao + Recall} \tag{4}$$

Para a implementação destas métricas a biblioteca *Sklearn Metrics* foi importada para o código do trabalho. Além disso, o *Framework AutoSkLearn* fornece, após sua execução, algumas outras métricas que se referem ao processamento, que são: quantidade de modelos iniciados, quantidade de modelos executados com sucesso, quantidade de modelos que excederam o tempo parametrizado, quantidade de modelos que excederam a memória e quantidade de modelos que geraram erros durante a execução.

5 RESULTADOS E DISCUSSÕES

Foram realizadas duas execuções do *framework AutoskLearn* sobre a base de dados, Pesquisa de Presos em estabelecimentos prisionais estaduais e federais - SIS-FCF (JUSTICE, 2004), com o objetivo de prever se um prisioneiro irá ou não quebrar alguma regra dentro do presídio. As Tabelas 6 e 7 apresentam os resultados encontrados após as execuções.

Na Tabela 6 é possível observar que a classe SIM apresentou maior precisão em relação à classe NÃO; ou seja, a taxa de falso positivo para a classe SIM foi menor do que a obtida para a classe NÃO. Assim, de todos os presos que não quebraram regras na prisão, os modelos classificaram mais de 30% como tendo quebrado regras. Contudo, a sensibilidade foi maior para a classe NÃO, obtendo uma média harmônica maior para esta classe. Veja, por exemplo, que no Teste 2 houve uma diferença de 4 pontos percentuais entre as duas classes. Nos outros testes, a diferença média foi de 2 pontos percentuais. Assim, para os testes realizados, os modelos estão acertando mais quem não quebrou regra na prisão. No entanto, a diferença de desempenho entre as classes é muito pequena e provavelmente não deve ter diferença significativa, caso seja realizado algum teste estatístico.

Tabela 6 – Análise de métricas dos Testes realizados.

Teste	Modelo	Classe	Precisão	Sensibilidade	F-Measure
1	SVM	SIM	0,74	0,66	0,70
		NÃO	0,68	0,76	0,72
		Média	**0,71**	**0,71**	**0,71**
2	Randon Gradiente Boosting	SIM	0,74	0,62	0,68
		NÃO	0,66	0,78	0,72
		Média	**0,71**	**0,70**	**0,70**
3	Gradiente Boosting	SIM	0,73	0,67	0,70
		NÃO	0,69	0,75	0,72
		Média	**0,71**	**0,71**	**0,71**
4	Randon Adaboost	SIM	0,73	0,68	0,70
		NÃO	0,69	0,75	0,72
		Média	**0,71**	**0,71**	**0,71**
5	Adaboost	SIM	0,73	0,68	0,70
		NÃO	0,69	0,74	0,71
		Média	**0,71**	**0,71**	**0,71**

A Tabela 7 apresenta as métricas, específicas, fornecidas pelo *framework Auto Sklearn* após a execução de cada teste. É possível analisar, na primeira métrica apresentada, a Acurácia, que representa a proximidade entre o valor obtido experimentalmente e o valor verdadeiro (CORK et al., 1983), o teste que obteve a melhor pontuação de acurácia foi o Teste 4, com diferença de um ponto percentual do segundo maior valor. As outras métricas fornecidas pelo *framework Auto Sklearn* apresentam os dados de desempenho da execução do *framework*; o Teste 3 foi o teste que executou a maior quantidade de ajustes sendo, neste teste, exclusivamente para o classificador *Gradient boosting*. O segundo teste que teve o maior número de execuções de ajuste foi o Teste 4, configurado com um tempo de execução maior, de dez horas, com a menor quantidade de parametrização possível, ou seja, foi previamente definido somente o tempo de execução.

Tabela 7 – *Sklearn Metrics* dos testes realizados

Métrica	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Acurácia	0,698	0,709	0,708	0,706	0,710
Quantidade modelos iniciados	153	308	1538	711	118
Quantidade modelos executados com sucesso	139	272	1115	629	112
Quantidade Modelos interrompidos por erro	5	5	15	15	0
Quantidade Modelos interrompidos por tempo	2	3	0	1	1
Quantidade Modelos interrompidos por memória	7	28	408	66	5

A Figura 10 apresenta como os parâmetros que foram automaticamente otimizados pelo *framework* no Teste 5, que foi o teste que apresentou maior acurácia. Os parâmetros foram ajustados nas fases de pré-processamento, escolha do modelo e ajustes dos hiperparâmetros do modelo escolhido.

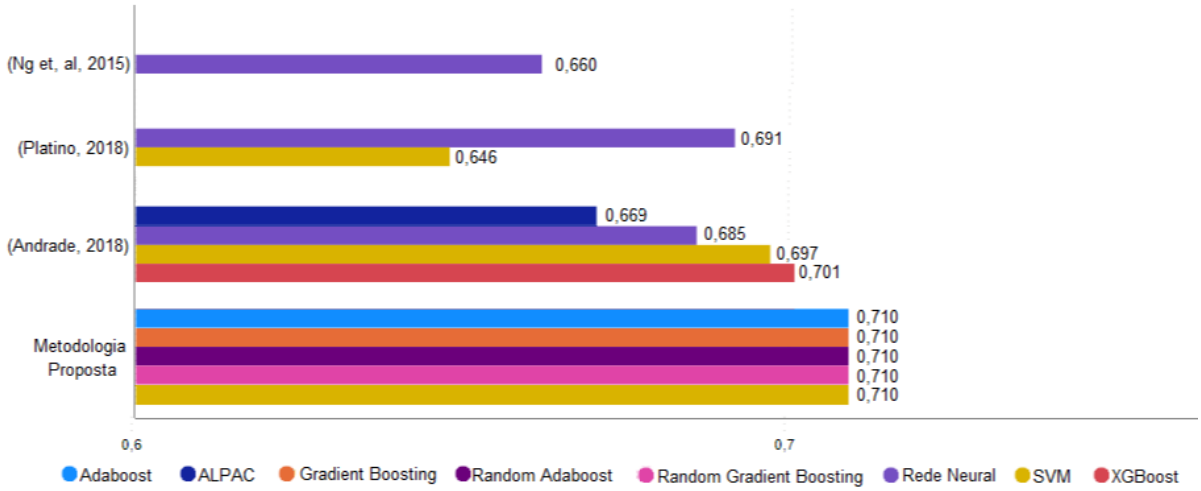
Figura 10 – Hiperparâmetros otimizados, Teste 5

Parâmetro	Valor
Estratégia de balanceamento	Nenhuma estratégia escolhida
Pré-processamento de dados categóricos	
Método de encoding	Nenhum método escolhido
Método coalescence	Nenhum método escolhido
Pré-processamento de dados numéricos	
Estrategia de imputação de dados	Média
Método de redimencionamento	quantile_transformer
Quantile transformer - Número de quantiles	1836
Quantile transformer - Distribuição	Normal
Modelo	
Modelo escolhido	adaboost
Algorítimo Adaboost	SAMME .R
Taxa de aprendizado	0.1740505434
Camada	1
Número de estimadores	479

Fonte: Elaborado pelo autor

É possível, ainda, realizar uma comparação entre os resultados obtidos por este trabalho e os trabalhos anteriores, utilizando-se a mesma base de dados. A comparação é realizada considerando-se os valores médios de *Precisão*, veja Figura 11.

Figura 11 – Comparação com os trabalhos anteriores, em relação ao valor da precisão



Fonte: Elaborado pelo autor

Analisando essa figura, é possível fazer alguns apontamentos. Os modelos gerados pelo aprendizado de máquina automático utilizados neste artigo foram equivalentes aos encontrados por Andrade (2018), já que a diferença é muito pequena (somente de 1 ou 2 pontos percentuais). No entanto, ressalta-se o potencial destas ferramentas de AutoML, visto que os trabalhos anteriores foram desenvolvidos ao longo de 2 anos, enquanto os testes realizados neste trabalho gastaram seis horas. Assim, acreditamos que estas ferramentas têm muito potencial para apoiar

os especialistas em ciência dos dados. Para resultados mais promissores, acreditamos que, um número maior de etapas de pré-processamento poderiam ser adicionadas aos *frameworks*.

6 CONSIDERAÇÕES FINAIS

Neste trabalho foram investigados os principais *frameworks* existentes para o aprendizado de máquina automático: MLBOX, Cloud AutoML, TPOT, H2O, Auto Keras e Auto Sklearn. Por meio deste levantamento, percebeu-se que esses *frameworks* são capazes de atuar, de forma diferente, nas etapas de leitura dos dados, limpeza dos dados, codificação/pré-processamento dos dados, seleção de modelo, otimização de hiperparâmetros e validação do modelo. Todas estas etapas são fundamentais para o sucesso na aplicação de um algoritmo de aprendizado de máquina, normalmente consideradas onerosas para a empresa e/ou o cientista de dados. Os problemas principais estão relacionados ao tempo e custo altos despendidos nas tarefas de ajuste da base de dados, escolha do modelo e otimização do mesmo por meio dos melhores parâmetros.

O *framework Auto Sklearn* foi considerado um dos *frameworks* mais promissores por possuir um grande número de etapas automáticas, ser de código aberto e ter condições de ajustar uma quantidade considerável de hiperparâmetros. Deste modo, durante a execução do trabalho, o *Auto Sklearn* foi aplicado a uma base de dados de presos em estabelecimentos prisionais estaduais e federais dos Estados Unidos, com o objetivo de predizer se o preso quebrará ou não a regra na prisão.

Ao comparar com trabalhos anteriores como os de Andrade (2018) e Piantino (2018), os resultados obtidos demonstram ser consideravelmente medianos. A melhora significativa dos resultados, que era almejada, não foi conquistada e o resultado geral ainda precisará ser melhorado. Entretanto, mesmo considerado mediano, o resultado obtido chama a atenção devido ao grande ganho de tempo na execução do modelo, uma vez que todo o modelo foi otimizado e executado durante um período máximo de 6 horas, tempo muito inferior aos quase dois anos gastos por Andrade (2018) e Piantino (2018).

Para trabalhos futuros sugere-se um empenho em melhorar os resultados avaliados, explorando as etapas de pré-processamento disponíveis na ferramenta. Melhores resultados consequentemente gerariam um maior conhecimento acerca dos perfis de possíveis infratores dentro dos presídios americanos. Com esse conhecimento, ações que visam melhorar e adequar o ambiente prisional poderão ser tomadas.

Referências

ANDRADE, Henrique Vasconcelos de. **Extração de conhecimento de métodos de Caixa Preta - Uma análise a partir de uma base de dados de presos americanos**. 2018. Dissertação (Artigo de Graduação) — Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte.

AUTOML. **TPOT**. 2019. Disponível em: <http://automl.info/tpot/>. Acesso em: 26 out. 2019.

BALAJI, Adithya; ALLEN, Alexander. Benchmarking automatic machine learning frameworks. **arXiv preprint arXiv:1808.06492**, 2018.

BATISTA, Gustavo Enrique de Almeida Prado et al. **Pré-processamento de dados em aprendizado de máquina supervisionado**. 2003. Tese (Doutorado) — Universidade de São Paulo.

BERGSTRA, James; BENGIO, Yoshua. Random search for hyper-parameter optimization. **Journal of machine learning research**, v. 13, n. Feb, p. 281–305, 2012.

CORK, Randall C; VAUGHAN, Robert W; HUMPHREY, Linda S. Precision and accuracy of intraoperative temperature monitoring. **Anesthesia and analgesia**, v. 62, n. 2, p. 211–214, 1983.

DATA LAB. **Autokeras**. 2019. Disponível em: <https://autokeras.com/>. Acesso em: 25 aph. 2019.

FACELI, Kattil et al. **Inteligência artificial : uma abordagem de aprendizado de máquina**. Rio de Janeito, RJ: LTC, 2011.

FEURER, Matthias et al. **Auto-sklearn: Efficient and Robust Automated Machine Learning**. [S.l.]: AutoML, 2019.

GIJSBERS, Pieter et al. An open source automl benchmark. **arXiv preprint arXiv:1907.00909**, 2019.

GOOGLE. **automl**. 2018. Disponível em: <https://cloud.google.com/automl/docs/>. Acesso em: 25 aph. 2019.

H2O. **H20.ai**. 2017. Disponível em: <http://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html>. Acesso em: 26 out. 2019.

HALL, M. et al. The weka data mining software: an update. **ACM SIGKDD explorations newsletter**, v. 11(1), p. 10—-18, 2009.

IKEDA, Denis T. Seleção não-supervisionada de métodos de binarização para documentos históricos. **Departamento de Ciência da Computação, Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo**, 2011.

JIN, Haifeng; SONG, Qingquan; HU, Xia. Auto-keras: An efficient neural architecture search system. p. 1946–1956, 2019.

KOTTHOFF, Lars et al. Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka. **The Journal of Machine Learning Research**, JMLR. org, v. 18, n. 1, p. 826–830, 2017.

LIASHCHYNSKYI, Petro; LIASHCHYNSKYI, Pavlo. Grid search, random search, genetic algorithm: A big comparison for nas. **arXiv preprint arXiv:1912.06059**, 2019.

MITCHELL, Tom M. **Machine Learning**. Ithaca, NY: : McGraw-Hill Science, 1997.

MONARD, Maria Carolina; BARANAUSKAS, José Augusto. Conceitos sobre aprendizado de máquina. **Sistemas inteligentes-Fundamentos e aplicações**, São Paulo, v. 1, n. 1, p. 1–32, 2003.

NGO, F. T.; R.GOVINDU; A.AGARWAL. Assessing the predictive utility of logistic regression, classification and regression tree, chi-squared automatic interaction detection, and neural network models in predicting inmate misconduct. **American Journal of Criminal Justice**, v. 40(1), p. 47–74, 2015.

OLSON, Randal S. et al. Evaluation of a tree-based pipeline optimization tool for automating data science. In: **Proceedings of the Genetic and Evolutionary Computation Conference 2016**. New York, NY, USA: ACM, 2016. (GECCO '16), p. 485–492. ISBN 978-1-4503-4206-3. Disponível em: <http://doi.acm.org/10.1145/2908812.2908918>.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

PIANTINO, C. C. P. **Application of artificial intelligence techniques inanalyzing inmate misconduct profiles**. 2018. Dissertação (Artigo de Graduação) — Pontifícia Universidade Católica de Minas Gerais, Belo Horizonte.

PINTO N., Doukhan D. DiCarlo J. J.; COX, D. D. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. **PLoS Computational Biology**, v. 5, n. 11, 2009.

ROMBLAY, Axel ARONIO DE. **MLBox**. 2019. Last accessed 14 November 2019. Disponível em: <https://mlbox.readthedocs.io/en/latest/>.

SHAWI, Radwa El; MAHER, Mohamed; SAKR, Sherif. **Automated Machine Learning: State-of-The-Art and Open Challenges**. 2019.

THORNTON, Chris et al. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In: **Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2013. p. 847–855.

YAO, Quanming et al. Taking human out of learning applications: A survey on automated machine learning. **arXiv preprint arXiv:1810.13306**, 2018.

ZÖLLER, Marc-André; HUBER, Marco F. Benchmark and survey of automated machine learning frameworks.