Universidade Estadual de Londrina

SAULO MARTIELLO MASTELINI

DSTARS:
UMA NOVA ABORDAGEM PARA REGRESSÃO MULTI-TARGET

Londrina
2018

SAULO MARTIELLO MASTELINI

DSTARS:
UMA NOVA ABORDAGEM PARA REGRESSÃO MULTI-TARGET

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Sylvio Barbon Junior.

Londrina
2018

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

Mastelini, Saulo Martiello.
DSTARS: uma nova abordagem para regressão multi-target / Saulo Martiello Mastelini. - Londrina, 2018.
104 f. : il.

Orientador: Sylvio Barbon Jr..
Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2018.
Inclui bibliografia.

1. Apredizado de Máquina - Tese. 2. Regressão multi-target - Tese. 3. Regressão multi-output - Tese. I. Barbon Jr., Sylvio. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. III. Título.

SAULO MARTIELLO MASTELINI

DSTARS:
UMA NOVA ABORDAGEM PARA REGRESSÃO MULTI-TARGET

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

Orientador: Prof. Dr. Sylvio Barbon Junior
Universidade Estadual de Londrina - UEL

Prof. Dr. Bruno Bogaz Zarpelão
Universidade Estadual de Londrina - UEL

Prof. Dr. Daniel dos Santos Kaster
Universidade Estadual de Londrina - UEL

Prof. Dr. Taufik Abrão
Universidade Estadual de Londrina - UEL

Londrina, 27 de Fevereiro de 2018.

Dedico esse trabalho a todos que veem na tecnologia o caminho para um mundo melhor.

AGRADECIMENTOS

Gostaria primeiramente de agradecer a Deus, por sempre ter me mantido, de uma forma ou outra, no caminho que considero correto e prover todas as condições e circustâncias para que esse trabalho se tornasse possível.

Em segundo lugar, agradeço à minha família, por sempre me guiar, aconselhar, cuidar e dar as condições para que eu pudesse chegar onde estou. Sei que os sacrifícios e decisões difíceis foram muitas nessas mais de duas décadas, mas sem isso de forma alguma teria sido possível. Deixo de forma especial meu agradecimento para minha mãe Vera Lucia Martiello Mastelini e meu pai Antonio Edson Mastelini, vocês são essenciais na caminhada. Agradeço também minha irmã Ana Paula Martiello Mastelini por ser a minha conselheira e a amiga desde sempre, e minha avó Geni Luzia Vacario Martiello por sempre sonhar com uma vida melhor para seus filhos e netos.

Agradeço também pelos meus amigos, companheiros nos momentos bons e difíceis. Sempre foram uma presença constante e confiável, sem vocês a luta teria sido muito difícil.

Gostaria também de agradecer a CAPES pelo suporte financeiro durante o decorrer da minha pesquisa.

E, por fim, gostaria agradecer ao professor Dr. Sylvio Barbon Jr., que após quase seis anos de parceira passou a ser mais que um orientador, e sim um grande amigo.

"But the game never ends
when your whole world depends
On the turn of a friendly card"
(The Alan Parsons Project,
The turn of a friendly card)

MASTELINI, Saulo Martiello. **DSTARS:** uma nova abordagem para regressão multi-target. 2018. 104 f. Dissertação (Mestrado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina, 2018.

RESUMO

Tradicionalmente, o aprendizado de máquina lida com problemas contendo uma única variável de saída, categórica ou contínua, resultando, dessa forma, em problemas de classificação e regressão, respectivamente. No entanto, muitos problemas da vida real associam um mesmo conjunto de variáveis descritivas a múltiplas variáveis de saída. Quando tais variáveis de saída, ou alvos, são contínuos, a tarefa preditiva é chamada de regressão *multitarget* (MT). Uma opção é tratar cada alvo como um problema separado, empregando uma abordagem *single-target* (ST). Todavia, nesse tipo de problema, além das relações entre as variáveis de entrada e saída, podem existir inter-dependências entre os alvos. Assim, a exploração desses relacionamentos pode levar ao aumento de desempenho preditivo. Nesse trabalho um novo método, chamado *Deep Structure for Tracking Asynchronous Regressor Stacking* (DSTARS), é proposto. A ideia principal do DSTARS é empregar múltiplas camadas de regressores empilhados para fornecer aproximações dos alvos, sendo que estas são utilizadas como atributos descritivos adicionais para o problema tratado. No método proposto, um número diferente de camadas de regressores é determinado para cada alvo de forma a explorar diferentes níveis de dependências estatísticas entre as saídas. Adicionalmente, o DSTARS explicitamente mensura as dependências estatísticas entre os alvos com uma métrica não-linear de importância de variáveis e, dessa forma, trata separadamente as saídas sem correlação como um problema ST. O método proposto foi avaliado em dezoito bases de dados de *benchmarking*, contra três abordagens da literatura para regressão MT, utilizando quatro técnicas de regressão. Os resultados obtidos demonstraram que o DSTARS foi capaz de obter resultados estatisticamente superiores às outras abordagens, obtendo os menores valores de erro preditivo na maioria dos casos avaliados.

Palavras-chave: Apredizado de Máquina. Regressão multi-target. Regressão multi-output.

MASTELINI, Saulo Martiello. **DSTARS:** a novel approach for multi-target regression. 2018. 104 p. Dissertation (Master's Degree in Science in Computer Science) – Universidade Estadual de Londrina, Londrina, 2018.

ABSTRACT

Machine learning traditionally deals with problems that present a single categorical or continuous output variable. Thus, it results in classification and regression tasks, respectively. However, many real-life problems associate a set of descriptive features to multiple output variables. When these outputs, or targets, are continuous the related predictive task is called multi-target regression (MTR). A possible approach to model these problems is to treat each target as a separate task, by employing a single-target (ST) strategy. Nevertheless, in MTR problems, besides the input-output relationships, inter-output dependencies may exist. So, the exploration of these inter-target dependencies could lead to better predictive performance. This work proposes a novel MTR method called Deep Structure for Tracking Asynchronous Regressor Stacking (DSTARS). The DSTARS' main idea is to employ multiple layers of stacked regressors to provide continuously improved targets' approximations. These estimates are used as additional descriptive attributes to the dealt problem. In the proposed method, each target has a different number of regressor layers to model different levels of statistical dependencies among the outputs. Additionally, DSTARS explicitly measures the statistical dependencies between the targets with a non-linear variable importance metric. Thus, uncorrelated targets are treated as independent ST tasks. The proposed method was evaluated on eighteen MTR benchmark datasets, against three approaches for MTR consolidated in the literature. Besides, four regression techniques were combined with the MTR methods. The obtained results showed that the DSTARS was statistically better than the other approaches and also obtained the smallest prediction errors in most of the cases.

Keywords: Machine Learning. Multi-target regression. Multi-output regression.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizado de Máquina
aRRMSE	*Average Relative Root Mean Squared Error*
CART	*Classification and Regression Tree*
CD	*Critical Distance*
DSTARS	*Deep Structure for Asyncronous Regressor Stacking*
ERC	*Ensemble of Regressor Chains*
FIRE	*Fitted Rules Ensemble*
k-NN	*k Nearest Neighbours*
MLP	*Multi-layer Perceptron*
MO	*Multi-output*
MORF	*Multi-objective Random Forest*
MSE	*Mean Square Error*
MT	*Multi-target*
MTR	*Multi-target regression*
MTRS	*Multi-target Regressor Stacking*
OOB	*Out-of-Bag*
OOBE	*Out-of-Bag Error*
PCA	*Principal Component Analysis*
PCT	*Predictive Clustering Trees*
PLSR	*Partial Least Squares Regression*
PR	*Performance* Relativa
RBM	*Restricted Boltzmann Machine*
RC	*Regressor Chain*
RF	*Random Forest*

RLC	*Random Linear Combinations*
RMSE	*Root Mean Squared Error*
RNA	Rede Neural Artificial
RRMSE	*Relative Root Mean Squared Error*
RV	Redução de Variância
ST	*Single-target*
SVM	*Support Vector Machine*
SVR	*Support Vector Regression*
XGBoost	*Extreme Gradient Boosting*

LISTA DE SÍMBOLOS

X	Atributos de entrada em problemas de AM
y	Variável de saída em problemas de AM de única saída
Y	Variáveis de saída em um problema MT
N	Número de instâncias em um problema de AM
m	Número de variáveis de entrada em um problema MT
d	Número de variáveis de saída em um problema MT
h	Modelo de AM supervisionado criado sobre as variáveis de entrada e saída
ϕ	Limiar de relevância para escolha de camadas de regressores no DSTARS
ε	Critério de convergência para o procedimento de busca pelo melhor número de camadas de regressores no DSTARS
F	Valores de filtragem resultantes da etapa de *Filtering* do DSTARS
T	Melhor disposição de camadas de regressores encontradas durante a etapa de *Tracking* do DSTARS

SUMÁRIO

1 INTRODUÇÃO

O ser humano constantemente aprende com experiências, erros e novas situações para as quais se expõe durante as mais variadas atividades. Nesse contexto, desenvolver métodos algorítmicos que também aprendam é um dos grandes desafios enfrentados pela área de Aprendizado de Máquina (AM) [1].

De fato, essa área de pesquisa computacional tem crescido no decorrer dos anos, tratando problemas variados que envolvem outras áreas, como a medicina, aspectos ambientais, engenharias de alimentos, civil e elétrica, dentre muitas outras áreas [1, 2, 3, 4, 5, 6]. Apesar de as áreas de pesquisa para as quais os esforços em AM se direcionam terem objetivos diversos, os construtos e mecanismos teóricos empregados nas resoluções tratam da descoberta e reconhecimento de padrões. A partir do momento que um conjunto descritivo de informações pode ser relacionado a uma resposta ou saída esperada, padrões devem existir no dado problema, permitindo a aplicação de ferramentas computacionais que simulem o pensamento, o processo decisório e aprendam com a experiência [1].

Quando os problemas modelados contam com uma saída ou resposta esperada para cada novo caso encontrado, eles são caracterizados como problemas classificatórios ou de regressão. A classificação se refere a atribuir para uma instância a ser avaliada uma categoria, à qual esta pertence, ou seja, uma classe. Em uma tarefa de regressão, as respostas assumem valores contínuos, numéricos, podendo representar diferentes grandezas. Problemas preditivos dessa natureza são chamados de *single-target* (ST), ou problemas de único alvo, resposta ou saída.

No entanto, em muitos dos desafios preditivos encontrados na vida real, o problema conta com múltiplas saídas relacionadas a um mesmo conjunto descritivo de atributos. Um exemplo para esse tipo de problema seria a predição do clima em intervalos de tempos futuros: dado um mesmo conjunto descritivo da situação atual do clima e de períodos anteriores, busca-se realizar previsões para diferentes datas no futuro. Claramente, trata-se de um problema que usa o mesmo conjunto para realizar predições futuras em diferentes intervalos. Além disso, a ocorrência de um fenômeno climático no primeiro dia para o qual a análise está sendo feita, provavelmente influenciará na situação climática dos dias posteriores. Portanto, além das múltiplas saídas, no dado exemplo, o valor de uma saída se relaciona com os outros, havendo uma dependência estatística entre as informações [4, 5].

De fato, problemas com dependência, correlações ou relacionamentos estatísticos entre as saídas podem ocorrer nos mais diversos campos, como em aplicações que realizam a predição de vendas em intervalos de tempo determinados, a inferência da composição química de solos e da água, a automação de processos industriais e a inferência de carga

energética [3, 4, 5]. Todas as tarefas preditivas apresentadas contam com múltiplas variáveis de saída contínuas. O ramo do aprendizado de máquina que estuda problemas desse tipo é chamado de regressão *multi-target* (MT) [4, 5]. A utilização de soluções MT em detrimento de realizar a predição individual de cada saída traz a vantagem de compreender o problema globalmente, modelando a dependência existente entre as múltiplas variáveis de saída existentes e, assim, melhor descrevendo a tarefa. Além disso, técnicas MT reduzem o *sobreajuste* aos dados de treinamento, quando comparados com uma coleção de modelos ST [7, 4]. Algumas das soluções MT tem a sua origem na área de classificação *multi-label*, que estuda problemas preditivos com múltiplas saídas binárias. Algumas técnicas foram adaptadas dessa área de pesquisa, incluindo os métodos *Multi-Target Regressor Stacking* (MTRS) e *Ensemble of Regressor Chains* (ERC), que obtiveram resultados favoráveis quando comparados com outras técnicas existentes [5]. Tais métodos empregam algoritmos de regressão ST e a manipulação dos dados de entrada para a modelagem das dependências estatísticas entre as saídas.

Soluções como as apresentadas utilizam manipulações no conjunto descritivo da tarefa, de modo a permitir que técnicas tradicionais e amplamente conhecidas para resolução de problemas com uma única saída possam ser utilizadas para modelar uma tarefa *MT*. Denomina-se o algoritmo utilizado para regressão em conjunto com a modelagem *MT* como *regressor base*. As relações existentes entre as saídas são mapeadas e introduzidas na solução construída de forma indireta, ou seja, sem a necessidade de criação de um algoritmo específico para gerar múltiplas saídas e explorar as dependências existentes entre estas [5, 8]. A criação de algoritmos que criam um único modelo preditivo para tratar o problema de regressão MT de uma única vez foi realizada utilizando diferentes abordagens [9, 10, 11, 3, 12, 13, 14]. No entanto, tal tarefa não é simples, demandando modificações profundas em técnicas conhecidas, ou a completa criação de um novo método. O emprego de regressores base combinados com um método MT traz maior flexibilidade para a exploração de diferentes estratégias de regressão e, de fato, tem apresentado resultados superiores à utilização de um único modelo preditivo [5, 8].

Dentre as soluções MT mencionadas, dois métodos que se utilizam da manipulação dos dados e regressores de uma única saída foram propostas recentemente [5], utilizando diferentes estratégias para modelagem das relações entre as variáveis de saída: o MTRS e ERC, como já mencionados anteriormente. As duas abordagens deram origem a trabalhos posteriores, que realizaram adaptações em sua formulação original [8, 15, 16, 17]. Os dois métodos partem da ideia de utilizar predições de cada uma das saídas, obtidas por regressores ST, como atributos descritivos adicionais, que são utilizados por outros modelos preditivos. Esse procedimento é denominado empilhamento de regressores. O MTRS utiliza como passo inicial a criação de modelos para predição individual de cada variável de saída, obtendo aproximações das respostas desejadas. Essas aproximações entram em uma segunda rodada preditiva como atributos descritivos adicionais para o problema,

onde novamente modelos preditores para cada alvo são induzidos, representando as saídas finais do esquema.

Partindo do pressuposto do não-conhecimento prévio do problema tratado e, dessa forma, das características das correlações existentes entre suas múltiplas saídas, o ERC utiliza um conjunto de cadeias de preditores baseados nas diferentes permutações possíveis entre as variáveis de saída da tarefa tratada. Tais preditores são construídos seguindo a ordem definida por essas sequências. Nesse contexto, uma cadeia consiste em modelos sucessivamente induzidos, que se utilizam das predições (aproximações) dos modelos anteriores no conjunto como atributos descritivos extras, assim como no MTRS. Após a construção dos múltiplos preditores, a saída gerada para cada variável é definida como a média entre as saídas de todos os regressores que foram treinados para predizer o alvo em questão, independente de sua posição nas cadeias (*chains*).

Apesar de representarem as soluções com alguns dos melhores desempenhos preditivos em problemas de regressão MT, o MTRS e o ERC, assim como outras soluções na área, apresentam alguns problemas. Esses métodos não pressupõem a possibilidade de existirem diferentes níveis de dependência estatística entre as saídas. Tampouco as soluções existentes preveem que alguns dos alvos possam não ter relações com ou outros, enquanto os outros as possuem. De fato, as abordagens utilizadas pelos métodos mencionados, para inserir as correlações entre as respostas na modelagem de problemas MT, são bastante limitadas. O MTRS utiliza um conjunto fixo de preditores, que utilizam aproximações de todas as variáveis alvo como atributos adicionais, sem avaliar se essas informações são realmente relevantes para a resolução do problema. Além disso, o MTRS não considera que alguns dos alvos, os mais dependentes dos outros, possam necessitar de mais iterações utilizando as aproximações das respostas como fonte de informação, para serem preditos com menores taxas de erros. O ERC, por sua vez, aleatoriamente explora diferentes ordenações das respostas para composição de modelos preditivos que empilham as predições dos regressores precedentes nas permutações definidas. Essa estratégia pode tanto deixar de considerar algumas correlações, quanto assumir relações inexistes, devido à aleatoriedade da modelagem. Os métodos existentes não apresentam mecanismos para exploração das características individuais de cada alvo, e consequentemente, das especificidades dos problemas preditivos tratados.

Esse trabalho apresenta um novo método para problemas de regressão MT chamado DSTARS (*Deep Structure for Tracking Assyncronous Regressors Stacking*), ou estrutura profunda para rastreamento assíncrono de empilhamento de regressores, em português. O método proposto utiliza um mecanismo de empilhamento de regressores similar ao MTRS, se baseando na inserção de aproximações das variáveis de saída como atributos descritivos extras. No entanto, o DSTARS emprega um número adaptativo de modelos de regressão empilhados para cada um dos alvos. Vários modelos, dispostos em camadas, são

utilizados para a composição de melhores estimativas das variáveis de saída ou *targets*, continuamente aprimorando as predições através da utilização das aproximações anteriores. O processo é realizado de forma independente para cada variável de saída, explorando a influência entre estas (caso exista). A hipótese para o DSTARS é que o emprego de um número diferente e potencialmente grande de camadas de regressores para cada alvo pode levar a desempenhos preditivos superiores, bem como explorar os diferentes níveis de dependências estatísticas que podem existir entre os alvos.

Contrastando com outros trabalhos na área, o método proposto utiliza as próprias características do problema tratado para determinar a melhor disposição de camadas de regressores para cada alvo. Assim, para alvos com maior dependência estatística um maior número de camadas de regressores é induzido e vice-versa. Assim, a própria composição de camadas do modelo gerado oferece pistas da dependência existente entre as saídas do problema, oferecendo um mecanismo para melhor interpretar as características da tarefa tratada. Por fim, o DSTARS explicitamente mensura as dependências estatísticas existentes entre os alvos através de uma métrica não-linear de importância baseada no algoritmo da *Random Forest*. Dessa forma, se um determinado alvo não possui dependência estatística com os outros, este é tratado como um problema ST separado, não utilizando estimativas dos outros como atributos descritivos adicionais. Além disso, considerando-se um dos alvos do problema preditivo em questão, somente as variáveis de saída que o influenciam são empregadas como fonte adicional de informação para a indução de regressores.

O novo método para regressão MT proposto nesse trabalho é comparado com três outras abordagens, sendo uma delas a regressão individual de cada saída (ST) e duas técnicas MT baseadas na expansão dos atributos originais com aproximações das variáveis alvo (MTRS e ERC). Os métodos foram executados em dezoito bases de dados de *benchmark*. Além disso, quatro técnicas de regressão base foram empregadas, sendo estas *Random Forest* (RF), *Support Vector Machine* (SVM), *Extreme Gradient Boosting* (XGBoost) e *Classification and Regression Tree* (CART), de forma a investigar o desempenho obtido por diferentes famílias de algoritmos preditivos. Os resultados obtidos mostraram que o DSTARS foi estatisticamente superior às outras abordagens e obteve os menores valores de erro preditivo na maioria dos casos.

O trabalho está organizado da seguinte forma: o Capítulo 2 apresenta a fundamentação teórica em Aprendizado de Máquina necessária para a compreensão do trabalho. No Capítulo 3 os trabalhos relacionados são apresentados, destacando suas vantagens e desvantagens, bem como as melhoria e contribuições oferecidas pelo DSTARS. No Capítulo 4 a técnica DSTARS é descrita de forma detalhada. Os materiais e métodos empregados para a realização dos experimentos são descritos no Capítulo 5. Os resultados obtidos, bem como a discussão destes é realizada no Capítulo 6. Em seguida, as considerações finais do trabalho e os próximos passos a serem realizados são discutidos no Capítulo 7.

Por fim, as referências bibliográficas utilizadas para embasamento do trabalho são apresentadas, seguidas por um Apêndice contendo informações extras acerca dos resultados obtidos.

2 APRENDIZADO DE MÁQUINA

A área de pesquisa em Aprendizado de Máquina (AM) trata do estudo de como desenvolver programas que automaticamente podem melhorar com a experiência [1]. Por essa razão, o estudo de AM engloba aspectos provenientes de várias áreas de conhecimento, como a estatística, inteligência artificial, filosofia, teoria da informação, biologia, ciência cognitiva, complexidade computacional e teoria do controle [1]. Além disso, a busca por rotinas que possam melhorar com a experiência, isto é, aprender, demanda a definição de meios para o aprendizado, formas de representação matemática dos problemas, métricas de desempenho, modelos de aprendizado, entre outros aspectos, de modo a serem interpretáveis e manipuláveis pela máquina.

Em relação às formas de aprendizado, tradicionalmente três estratégias de ensino podem ser adotadas: o aprendizado supervisionado, não-supervisionado e semi-supervisionado [1, 18]. No aprendizado supervisionado, existe a figura de um especialista, técnico ou professor que sabe e fornece as respostas corretas para um determinado problema. Dessa forma, o algoritmo de aprendizado aprende a associar conjuntos de padrões com a resposta esperada. Além disso, os algoritmos de aprendizado supervisionado não devem simplesmente decorar os exemplos para os quais foram treinados, e sim generalizar conhecimento acerca do problema tratado a partir do subconjunto ao qual obtiveram acesso para aprendizagem. Em problemas de aprendizado supervisionado, comumente, uma base de dados cujas respostas já estão assinaladas é fornecida para um algoritmo de AM, que utiliza esse conjunto para aprender.

Uma segunda possibilidade é o aprendizado não-supervisionado: nesse caso, não se conhece qual a saída desejada, nem mesmo um comportamento esperado; os dados descritivos do problema são conhecidos e, cabe à técnica de AM extrair padrões, inferir comportamentos, agrupar os dados em conjuntos similares, entre outros aspectos. Dentre as técnicas de aprendizado não-supervisionado se destacam os algoritmos de clusterização, como o k-médias (*k-Means*) e o *dbscan*, e a extração de características, como a Análise de Componentes Principais (*Principal Component Analysis* - PCA), dentre outros [19].

Por fim, no aprendizado semi-supervisionado parte da saída esperada é conhecida e esta informação é utilizada para caracterizar comportamentos similares ao conjunto conhecido, bem como determinar novos grupos disjuntos do conjunto de informação disponível. Dentre as soluções para tais problemas, técnicas derivadas de algoritmos de aprendizado supervisionado podem ser utilizadas [20, 21], além de outras abordagens, como o uso de Máquinas de Boltzmann Restritas (*Restricted Boltzmann Machine* - RBM) [18].

O método proposto nesse trabalho se relaciona ao campo de pesquisa de aprendi-

zado supervisionado de máquina. De modo formal, um problema de aprendizado supervisionado é definido como a tarefa de se encontrar uma função ou conjunto de funções h que relacionem um conjunto de m variáveis de entrada, ou variáveis explanatórias X, com uma variável de saída y. Para tal, N exemplos conhecidos do problema tratado são fornecidos ao algoritmos de AM que, por sua vez, produz um modelo preditivo, representados por h. A saída, y, frequentemente também é referida como o alvo ou rótulo da tarefa preditiva. A Equação 2.1 apresenta a expressão relacionada à definição de um modelo preditivo em um problema de AM supervisionado.

$$h : X_{1..m} \rightarrow y \tag{2.1}$$

Durante a busca por h objetiva-se minimizar o erro preditivo associado ao modelo em construção, de forma a levar este a se adaptar e aprender as características do problema em questão. Assim, espera-se que o modelo gerado seja capaz de predizer os valores de y com o menor índice de erros possível, de acordo com uma métrica de erros arbitrária escolhida. Além disso, objetiva-se que as funções em h sejam o mais simples o possível, de modo que estas não se tornem sobreajustadas aos exemplos disponíveis [22], ou em outras palavras, muito atreladas aos exemplos de treinamento. Caso o sobreajuste aconteça, o modelo preditivo tende a se comportar de forma insatisfatória em instâncias da tarefa preditiva para as quais não foi previamente apresentado. Esse fato ocorre porque o preditor não aprendeu a generalizar conhecimento e sim decorou os casos aos quais foi apresentado. Para garantir que não ocorra sobreajuste diferentes estratégias de avaliação de modelos de AM são utilizadas, como apresentado na Seção 2.2. Por outro lado, se o modelo preditivo for demasiadamente genérico o conhecimento e os padrões relativos ao problema tratado não serão extraídos, ocasionando o fenômeno chamado subajuste. Busca-se em uma tarefa de predição por AM, portanto, encontrar um ponto de equilíbrio entre o subajuste e o sobreajuste.

De acordo com o tipo dos elementos em y, dois tipos de tarefas preditivas se resultam, sendo estas os problemas de classificação e regressão. Em tarefas de classificação, uma única saída categórica e não-ordenada é o alvo da predição. Em problemas de regressão, y assume valores numéricos contínuos ou discretos ordenados. O método proposto neste trabalho foi criado para tratar primariamente problemas de regressão. Quando se considera apenas uma variável de saída, também chamada de alvo (*target*) ou rótulo, os problemas tratados podem ser referidos como tarefas de predição de único alvo, ou tarefas *single-target* (ST).

Variadas bases teóricas são empregadas na formulação de algoritmos de aprendizado supervisionado [1, 3, 4, 5], resultando em variadas características, vantagens e desvantagens, bem como em diferentes custos computacionais, que são relativos ao número de instâncias utilizadas para aprendizagem no problema tratado [2]. Nesse trabalho

quatro técnicas de AM para regressão foram escolhidas: Máquina de Vetores de Suporte (*Support Vector Machine* - SVM), Árvore de Classificação e Regressão (*Classification and Regression Tree* - CART), Floresta Aleatória (*Random Forest* - RF) e Aumento Extremo de Gradiente (*Extreme Gradient Boosting* - XGBoost). As técnicas mencionadas são apresentadas na próxima seção.

2.1 Algoritmos de aprendizado supervisionado

Diferentes abordagens para criação de algoritmos de aprendizado supervisionado têm sido utilizadas, empregando variadas bases teóricas [23]. Estas incluem o emprego de técnicas simples, como o *k-vizinhos mais próximos* (*k-Nearest Neighbours* - k-NN), onde não se cria um modelo preditivo (tratando-se de uma técnica preguiçosa, ou *lazy*, como é comumente referida em inglês), e sim compara-se uma nova instância com as k amostras mais próximas a esta, sendo que os exemplos comparadas são previamente conhecidos. Por fim, no k-NN, infere-se uma resposta ao novo caso analisado a partir dos exemplos que se situam mais próximos a esse [24].

Outras metodologias utilizam a simulação de neurônios cerebrais para criação de Redes Neurais Artificiais (RNA), aproximando a forma como se dão as sinapses no cérebro [2, 3]. Um exemplo dessa categoria de técnicas são as Redes Neurais Perceptron com Múltiplas Camadas (*Multilayer Perceptron* - MLP) [2].

Os algoritmos baseados em *kernel* aplicam uma transformação sobre o conjunto de entrada, aumentando a dimensionalidade e buscando melhorar a separação das informações. As árvores de decisão separam recursivamente o conjunto de dados em partições cada vez menores, determinado fronteiras de decisão com grau decrescente de importância. Por fim, métodos *ensemble* (ou métodos baseados em conjuntos de preditores) partem do pressuposto de utilizarem múltiplos preditores mais simples de forma coordenada, para melhor capturar a variabilidade dos dados e aumentar a capacidade de generalização de conhecimento nos problemas tratados.

Duas estratégias principais têm sido aplicadas para a composição dos conjuntos de preditores: *Bagging* e *Boosting*. Na primeira, os modelos de AM são induzidos de forma paralela e independente, utilizando diferentes subconjuntos da base de dados de treinamento, tipicamente compostos por amostragem com reposição [25]. No *Boosting*, os preditores são construídos sequencialmente, objetivando diminuir o erro residual deixado pelos modelos induzidos anteriormente. Em ambas estratégias, todos os classificadores/regressores do conjunto treinado podem contribuir para a resposta final obtida.

Independente da estratégia adotada para a criação de uma algoritmo de AM supervisionado, além dos conjuntos de variáveis de entrada e a saída esperada, requerida para o treinamento dos modelos preditivos resultantes, outros atributos referentes à técnica utili-

zada podem ser passados pelo usuário. Tais parâmetros de configuração dos algoritmos de AM são referidos como hiper-parâmetros e, dependendo da técnica de AM considerada, influenciam grandemente no desempenho preditivo alcançado pelo preditor treinado [2]. Em muitos problemas de AM é comum a realização de uma etapa de ajuste (*tuning*) dos hiper-parâmetros dos algoritmos utilizados para o problema tratado, buscando assim, melhor ajustar ajustá-los à tarefa tratada e, consequentemente, diminuir os erros preditivos obtidos.

Nesse trabalho, tendo em vista que objetivou-se realizar uma comparação entre múltiplos métodos MT, todas as técnicas de regressão escolhidas utilizaram seus conjuntos de hiper-parâmetros padrão, de forma a alcançar uma comparação mais igualitária dentre todos os métodos considerados. As características principais de cada uma das técnicas de regressão escolhidas, bem como suas vantagens e desvantagens são apresentadas a seguir.

2.1.1 Support Vector Machine

A SVM é um algoritmo de aprendizado de máquina primeiramente proposto por Vapnik [26], pertencendo à classe dos métodos baseados em *kernel*. A ideia principal da técnica é encontrar o hiperplano (plano n-dimensional) que melhor separa os dados em determinadas classes. Dessa forma, as SVMs buscam maximizar a margem de separação existente entre as instâncias que se situam nas fronteiras de separação do dado problema. Tais instâncias são denominadas de Vetores de Suporte. Por se buscar maximizar as fronteiras de separação entre classes enquanto minimizando o erro classificatório, as SVMs estão relacionadas à resolução de um problema de otimização quadrático, que é relativo a encontrar o hiperplano de separação.

Para o caso de problemas não-lineares, a tarefa é tratada manipulando-se o espaço de entrada através de funções *kernel*: os dados de entrada (não linearmente separáveis) são mapeados para um espaço transformado, geralmente, de maior dimensão, onde estes podem ser linearmente separados [27]. Por essa característica as SVMs são consideradas métodos baseados em *kernel* [27, 4]. Como exemplos de funções *kernel*, frequentemente funções polinomiais de grau arbitrário e funções de base radial são empregadas [27].

A base teórica relacionada aos Vetores de Suporte pode ser estendida para a análise não-linear de regressão, utilizando a mesma metodologia baseada em *kernel* [27]. Para a regressão, após a transformação do espaço de entrada, um hiperplano é ajustado nos dados transformados objetivando minimizar uma função de erro escolhida, calculada entre os dados reais e preditos. Durante esse passo de otimização, objetiva-se também que a função de regressão escolhida seja a mais próxima possível da linear.

As SVMs têm se mostrado hábeis e flexíveis para tratar problemas preditivos diversos [28, 27, 23]. Devido a sua característica de maximização da margem de separabilidade dos dados, a capacidade de generalização também é aumentada, gerando resultados

acurados. Esse algoritmo foi projetado tendo em vista a manipulação de problemas de alta dimensionalidade, o que é facilitado por utilizar um único hiperplano de decisão, facilmente adaptável para qualquer dimensionalidade. Através da escolha correta de uma função de transformação (*kernel*), é possível atingir resultados muito similares às técnicas *ensemble*, consideradas o estado-da-arte [23].

Por outro lado, a escolha dos hiper-parâmetros e a função *kernel* são fatores impactantes no desempenho dos modelos gerados, sendo que uma decisão adequada depende muitas vezes do conhecimento técnico do utilizador e a experimentação. A SVM necessita da parametrização de elementos que alteram sensivelmente o resultado final obtido. Além da função *kernel*, a função de erro a ser empregada para minimização e um parâmetro de regularização devem ser definidos. O último é decisivo para determinar se o modelo gerado generalizará a compreensão do problema tratado ou se ficará sobreajustado aos dados de treinamento.

O valor de regularização determina um critério de balanceamento entre maximizar a separação dos dados e permitir certos níveis de erro. Em problemas não-lineares, comumente um certo nível de erro preditivo deve ser aceito para a definição satisfatória de um hiperplano de separação/regressão [27]. Esse fato está relacionado a um critério mais fraco de regularização (um valor mais baixo na parametrização), enquanto que, de forma contrária, valores de regularização muito altos tendem a levar ao sobreajuste do hiperplano gerado aos dados de treinamento. No entanto, se o valor de regularização for demasiadamente baixo, o modelo gerado não será capaz de aprender as características do problema, ocasionando um subajuste do hiperplano ao conjunto de treinamento.

Por fim, os modelos gerados pelas SVMs não são facilmente interpretáveis devido à complexidade envolvida nas transformações do espaço de entrada realizadas. Apesar de se tratar primariamente de um modelo linear de predição, as transformações aplicadas através das funções *kernel*, mencionadas anteriormente, adicionam não-linearidade à solução. Assim, as relações existentes entre os atributos explanatórios, capturadas pelo modelo preditivo gerado, tipicamente não são apresentadas de forma clara.

2.1.2 Classification and Regression Tree

As árvores de classificação e regressão são métodos de aprendizado de máquina que constroem modelos preditivos através do particionamento recursivo do espaço de entrada em subconjuntos menores de dados [29]. Um dos algoritmos para criação de árvores de decisão é a CART [30]. Modelos simples de predição são induzidos para cada partição, sendo que o particionamento resultante pode ser representado graficamente através de árvores de decisão. As árvores de classificação são projetadas para lidar com variáveis dependentes que tomam valores não-ordenados, sendo que o erro de predição é tomado em termos do custo do erro de classificação. As árvores de regressão lidam com problemas

onde as variáveis alvo assumem valores contínuos ou valores discretos ordenados, e o erro preditivo está associado a uma função de perda, que comumente é a diferença quadrática ou absoluta entre os valores observados e os preditos pelo modelo.

Os pontos de partição do espaço de dados são chamados de nós (não-terminais), enquanto que os locais onde uma resposta final é gerada são chamados de folhas (nós terminais). Cada caminho possível da raiz até uma folha pode ser visto como uma regra de decisão. Para a geração de uma árvore, a cada passo recursivo de particionamento, uma métrica de impureza dos dados nos nós da árvore é empregada. No algoritmo proposto por Breiman et al. [30], considerando-se tarefas de regressão, a métrica empregada é a redução da variância [30, 29]. Em problemas de regressão a ideia de impureza se relaciona com a variância observada nas respostas. Assim, as partições são feitas buscando-se reduzir cada vez mais a variância observada nos dados que pertencem a cada nó criado. A Equação 2.2 apresenta a métrica de Redução de Variância (RV) que é calculada pela CART sobre o conjunto de saída, representado por y.

$$RV(y, P) = \mathrm{Var(y)} - \sum_{p \in P} \frac{|p|}{|y|} \mathrm{Var}(p) \tag{2.2}$$

Na equação, `Var` representa o cálculo da variância entre elementos reais, enquanto P representa o conjunto de partições resultante da aplicação de um limiar, ou ponto de corte, sobre uma das variáveis de entrada do problema tratado. Comumente, todos os pontos possíveis para corte são testados a cada nó de decisão, considerando todas as variáveis e elementos pertencentes ao conjunto de entrada X. O valor que originar a maior redução na variância para o nó em questão será escolhido para originar os nós subsequentes. As partições escolhidas passarão a corresponder aos dados que serão avaliados posteriormente nos nós subsequentes da árvore, em outras palavras, os novos conjuntos X e y a serem avaliados em cada um dos nós gerados.

No algoritmo da CART, cada árvore é construída a partir da raiz até as folhas, em uma abordagem *top-down*, frequentemente, até o tamanho máximo possível. No entanto existe a possibilidade de aplicação de um procedimento de poda após a definição da estrutura decisória, ou mesmo durante sua construção. A poda é utilizada para evitar que a estrutura criada seja muito dependente do conjunto de treinamento utilizado, em outras palavras, sobreajustada a esses dados. A poda consiste em eliminar ramos, segundo um critério determinado, transformando-os em nós folhas e, assim, tornando a árvore menos profunda e específica quanto às regras de decisão geradas.

As árvores de decisão e, mais especificamente, o algoritmo CART constituem um modelo de aprendizado supervisionado de baixa complexidade, sendo viáveis para aplicações onde o tempo é um fator decisivo. Além disso, a CART é um algoritmo de fácil utilização por não contar com grande número de hiper-parâmetros a serem definidos. De

fato, apenas um valor de complexidade da árvore gerada deve ser escolhido, estando este relacionado à profundidade máxima do modelo resultante em razão do número de nós e folhas existentes. No entanto, apesar de contar apenas com um hiper-parâmetro, a escolha deste é um fator decisivo para o desempenho do classificador ou regressor gerado, definindo de forma última se o modelo se sobreajustará aos conjunto de treinamento, ou será demasiadamente genérico, não sendo capaz de descrever o problema tratado. Trata-se portanto da busca por um valor que venha a originar um modelo que seja capaz de balancear os dois extremos.

2.1.3 Random Forest

O algoritmo RF é uma abordagem *ensemble* proposta por Breiman [25]. A ideia principal da técnica é combinar várias árvores de decisão em uma floresta, o que nomeia a técnica. Assim, vários modelos de aprendizado de máquina com menor poder preditivo são combinados, de forma a gerar um único resultado.

Cada árvore na floresta é construída com o maior tamanho possível e sem poda, utilizando diferentes subconjuntos de atributos de entrada, selecionados aleatoriamente dentre todos os constituintes na base de dados. A quantidade de atributos a serem amostrados é definida pelo usuário, sendo o hiper-parâmetro da RF que mais influencia no resultado do método [25]. Além disso, cada novo preditor utiliza conjuntos diferentes de treinamento, com o mesmo número de exemplos da base de dados original, compostos através de amostragem com reposição (*bootstrap*) na estratégia de composição de *ensembles* denominada *Bagging* [25]. Na amostragem com reposição, uma instância pode ser escolhida mais de uma vez para compor o conjunto de treinamento. Busca-se através desses fatores de aleatorização abranger diferentes possibilidades e casos preditivos, diminuindo a influência de *outliers* e ruídos, bem como aumentando a capacidade de generalização do método.

Tradicionalmente, árvores do tipo CART são empregadas para a construção da floresta [25]. O número de árvores empregadas na floresta é definida pelo utilizador da técnica, sendo que essa decisão não é sensivelmente impactante no desempenho final do preditor gerado, quando comparada ao número de atributos para subamostragem [25]. Sendo novos dados submetidos ao conjunto, a saída final é definida como a agregação das resposta de todos os preditores (voto majoritário na classificação e valor médio na regressão).

As RFs são consideradas atualmente uma das técnicas de aprendizado de máquina supervisionado com maior poder preditivo [23], com grande capacidade de generalização. Devido aos critérios de aleatorização empregados, bem como o *Bagging*, é um algoritmo que consegue lidar bem com dados desbalanceados e ruidosos, comumente atingindo resultados muito acurados. Apesar de gerar modelos extensos, tendo em vista a quantidade

de preditores constituintes, também denominados *weak learners*, possui poucos hiperparâmetros a serem definidos, o que facilita a sua utilização e aplicação. Em geral, a utilização da parametrização padrão da RF costuma levar a desempenho preditivo elevado [25, 23].

2.1.4 Extreme Gradient Boosting

O XGBoost é um *framework* para *boosting* de árvores de decisão projetado para ser escalável [6]. Essa técnica tem chamado bastante atenção nos últimos anos, sendo empregada por diversas equipes vencedoras de competições envolvendo aprendizado de máquina [6]. Devido a procedimentos de otimização de algoritmos empregados, a solução provida pelo XGBoost tem apresentado resultados superiores aos de outras técnicas, sendo que os modelos gerados são processados de forma muito rápida.

Seguindo a metodologia de *boosting*, novos preditores (árvores) são adicionados buscando minimizar o erro total do *ensemble*. Termos de regularização são adicionados de forma a evitar o sobreajuste da solução ao conjunto de dados. Além disso, o *framework* emprega um fator de redução de importância de árvores individuais, similar ao fator de aprendizado de redes neurais, e a subamostragem de atributos, assim como na RF, como metodologias para redução de sobrejuste.

O XGBoost utiliza um algoritmo próprio para encontrar os melhores locais para realização dos cortes para a separação dos dados nas árvores treinadas. Em detrimento da abordagem tradicional que avalia todas as possibilidades de cortes dentre as instâncias, no mencionado *framework*, um procedimento aproximado é empregado. Esse critério de corte é baseado na distribuição estatística dos dados (quartis) e, quando utilizada a parametrização correta, leva a resultados muito próximos da abordagem tradicional, denominada gulosa [6].

Juntamente com a RF, outra abordagem *ensemble*, o XGBoost tem sido apontado como uma das técnicas com maior poder preditivo, o que justifica a sua utilização em uma ampla gama de problemas e tarefas [6]. Além de seu desempenho, a construção do *ensemble* é muito rápida devido aos critérios de otimização, paralelização e, mesmo construção das árvores, que são empregados. Trata-se de uma ferramenta com grandes possibilidades de configurações, customizações e extensões, potencialmente podendo alcançar melhor *performance* que outras técnicas de aprendizado de máquina [6].

2.2 Metodologias de avaliação para algoritmos de aprendizado supervisionado

Para verificar efetivamente o desempenho obtido por um classificador ou regressor treinado sobre uma base de conhecimento, algumas metodologias podem ser adotadas. Em

geral, deseja-se avaliar o nível de generalização do problema atingido, ou seja, garantir que para a existência de um novo conjunto de dados ainda não observado, o modelo induzido possa atingir desempenho satisfatório.

Como o referido novo conjunto de dados não é conhecido, opta-se por utilizar estratégias de amostragem para a composição de conjuntos disjuntos de treinamento e teste (o último também referido como conjunto de validação, em alguns casos), avaliando o desempenho obtido nessas análises como forma de aproximação do desempenho real do preditor. Comumente, quatro formas de análise são empregadas: *Holdout*, *Cross-validation* (validação cruzada), *Leave-One-Out* e amostragem *Bootstrap* [2].

O *Holdout* [2] consiste em separar o conjunto total de dados em duas partições, através de um limiar definido, uma de treinamento e outra de testes. Assim, um preditor é induzido na porção de treinamento e seu desempenho é avaliado sobre um conjunto de testes, o qual só contém dados não utilizados para o treinamento. Comumente, o corte empregado é de $\frac{2}{3}$ dos dados para treino e $\frac{1}{3}$ para teste; no entanto, esse valor pode ser alterado livremente. Outra abordagem comum é repetir esse processo várias vezes, tomando diferentes composições de treino e teste, através do embaralhamento dos dados no início de cada iteração, de modo a evitar a repetição de conjuntos. Quando utiliza-se um processo repetido de *holdout*, além da média de desempenho dentre todas as repetições, é costumeiro calcular o desvio padrão dos resultados obtidos, avaliando a estabilidade do preditor avaliado. Para o caso de problemas de classificação, é comum realizar uma amostragem estratificada, ou seja, garantir que no conjunto de treinamento uma proporção mínima de exemplos de cada classe esteja contida, evitando o desbalanceamento da base de treinamento e que o preditor se depare com uma classe para o qual não foi treinado para lidar, durante o teste.

Na abordagem de amostragem por validação cruzada (*Cross-validation*) [2], k partições ou dobras (*folds*) disjuntas são criadas no conjunto de dados. É comum se empregar um valor de $k = 10$. Em um processo iterativo, $k - 1$ dobras são utilizadas para a indução de preditores, enquanto que a partição deixada de fora do treinamento é utilizada para testes. Após todas as dobras serem empregadas como conjunto de testes, a média de desempenho obtida é tomada. Novamente, é comum embaralhar os dados da base utilizada, garantindo maior espalhamento dos dados utilizados. Quando o número de dobras utilizado é igual ao número de instâncias na base, a estratégia de amostragem passa a se chamar validação cruzada *Leave-One-Out*.

O processo de amostragem por *Bootstrap* [2] consiste em amostrar um conjunto de treinamento de tamanho arbitrariamente determinado, utilizando amostragem aleatória com reposição. Instâncias podem ser sorteadas nenhuma, uma ou mais vezes. Assim sendo, a base de treinamento resultante pode conter instâncias repetidas. O conjunto de teste é formado pelas amostras que não foram escolhidas para o treinamento. Esse procedimento

pode ser repetido várias vezes, de forma similar ao *holdout*, buscando explorar composições variadas e verificar a estabilidade dos preditores avaliados.

A Figura 1 ilustra as estratégias de amostragem apresentadas. Na variante *Holdout* apresentada 70% dos dados foram empregados para treinamento e 30% para teste. Um procedimento de validação cruzada com 10 dobras é também apresentado na figura. No exemplo a execução da primeira dobra é apresentada. Por fim, na variante *Bootstrap* o conjunto de treinamento apresentado foi definido com o mesmo tamanho da base original. O conjunto de testes nessa variante foi composto a partir dos elementos que não foram selecionados para o treinamento. No decorrer deste trabalho, a abordagem escolhida para amostragem das bases de dados empregadas foi a validação cruzada, levando em conta sua ampla utilização e balanceamento entre o custo e confiabilidade da avaliação [2, 23].

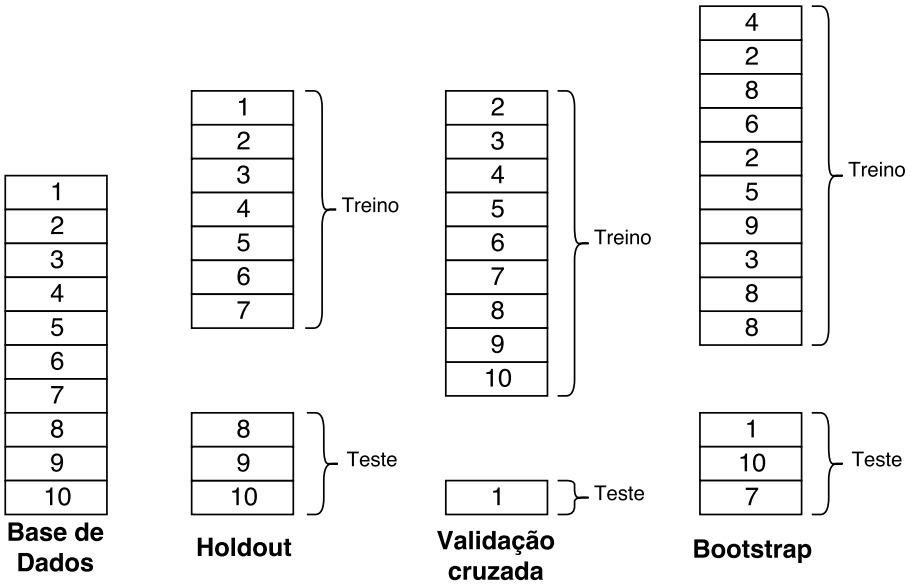

Figura 1 – Comparação entre as diferentes estratégias de amostragem.

2.3 Problemas preditivos com múltiplas saídas

Em muitos problemas de predição e aprendizado de máquina supervisionado, a tarefa preditiva conta com múltiplas saídas relacionadas ao mesmo conjunto de atributos descritivos, caracterizando um problema *multi-target* (MT). Formalmente, um problema preditivo de múltiplas saídas pode ser definido como a tarefa de encontrar uma função, ou um conjunto de funções h que sejam capazes de relacionar um grupo de m variáveis explanatórias X com um conjunto de d variáveis de saída Y [22]. A Equação 2.3 sumariza a tarefa preditiva relacionada à um problema MT.

$$h : X_{1..m} \rightarrow Y_{1..d} \tag{2.3}$$

De forma similar às tarefas ST, N instâncias do problema tratado são fornecidas para o algoritmo de aprendizagem para a geração dos modelos preditivos. Além disso, dependendo do tipo das variáveis contidas em Y, diferentes denominações são adotadas para os problemas de predição que são derivados, como afirma Kocev et al. [22]. Quando as respostas assumem valores binários, a tarefa derivada é denominada *classificação multi-label*. Em certos casos, as múltiplas respostas estão organizadas de forma hierárquica na forma de uma árvore ou grafo acíclico direcionado, o que resulta em problemas de *classificação hierárquica*. Esse tipo de problema também pode ser estendido para o caso *multi-label*, resultando em uma tarefa de *classificação hierárquica multi-label*. Além disso, um problema pode apresentar várias saídas categóricas que assumem múltiplos valores discretos e finitos, resultando em problemas de *classificação multi-target* ou classificação multi-dimensional. Por fim, quando as múltiplas respostas associadas a um problema assumem valores contínuos trata-se de um problema de *regressão multi-target*, que é o foco dessa pesquisa.

Problemas de regressão MT frequentemente são referenciados como problemas *multi-output* ou de regressão multivariada [9, 31, 4, 3, 5]. O método proposto, DSTARS, como já mencionado, foi projetado para lidar sumariamente com problemas de regressão MT; todavia, pode ser estendido para problemas de classificação *multi-label* trivialmente, utilizando regressores para os valores binários das saídas e aplicando limiares para binarizar as respostas obtidas.

Como forma de resolução, os problemas MT podem ser tratados de forma separada, induzindo um preditor para cada saída desejada. No entanto, em vários casos e situações da vida real as variáveis de saída possuem dependências estatísticas entre si [22, 4, 5, 8]. Em outras palavras, as grandezas referentes às saídas dos problemas preditivos estão correlacionadas e seus valores influenciam nas composições uns dos outros. De fato, o termo correlação se refere a relacionamentos funcionais entre grandezas, ou à medições dos níveis de relacionamento ou dependência entre as mesmas [32]. De forma similar, a ausência de correlação está relacionada à independência entre as grandezas comparadas. Em adição, as correlações podem ser de natureza linear e não-linear [32].

Técnicas capazes de modelar essas inter-dependências e inseri-las na formulação dos modelos preditivos tendem a gerar resultados superiores à predição individual de cada valor [10, 9, 22, 11, 31, 4, 3, 5, 8]. O método proposto se insere nesse contexto, buscando explorar as correlações entre os valores numéricos das variáveis respostas através do emprego de múltiplos regressores ST para cada uma das saídas, de forma dinâmica e sequencial, de forma a refinar a precisão das respostas obtidas através da utilização destas como atributos descritivos extras para o problema tratado. O uso de predições de modelos preditivos como fonte adicional de informação para problemas preditivos é referido na literatura como o empilhamento de preditores [33, 4, 5].

A descrição de trabalhos e técnicas relacionadas à área de regressão MT é feita no próximo capítulo, apresentando as soluções mais atuais e tradicionalmente utilizadas. Além disso, as vantagens, desvantagens e áreas abertas para pesquisa são discutidas.

3 TRABALHOS RELACIONADOS

A primeira alternativa para resolução de problemas regressão MT consiste em ignorar as possíveis dependências existentes entre os alvos, tratando cada saída de forma isolada. Dessa forma, um problema é tratado através de um conjunto de regressores ST, que compartilham o mesmo espaço de entrada mas lidam com diferentes saídas, de forma independente. Essa alternativa é denominada estratégia ST. Apesar de ser uma alternativa simples e não explorar as possíveis dependências existentes entre as variáveis alvo, em vários casos a estratégia ST foi capaz de superar soluções MT, dependendo do problema tratado [4, 5]. Essa estratégia é comumente empregada como um *baseline* de desempenho e será também utilizada como tal para a validação do método proposto. Em cenários em que existe dependência estatística entre as saídas, espera-se que técnicas MT alcancem maior desempenho preditivo, quando comparadas à estratégia ST. O fato da estratégia ST ter superado alguns métodos MT em alguns problemas é um indício da inabilidade das soluções MT existentes em modelar problemas onde podem existir diferentes níveis de dependências entre as saídas, ou mesmo não existirem tais dependências. O método proposto foi projetado tendo em vista esses problemas observados em trabalhos precedentes.

No artigo de Borchani et al. [4], um estudo extenso sobre a regressão MT é realizado, comparando diferentes métodos para resolução, métricas de avaliação, bases de *benchmarking*, entre outros aspectos. Segundo esse mesmo trabalho, tradicionalmente, problemas MT tem sido resolvidos através de duas abordagens principais: a **adaptação de algoritmos** de regressão e a **transformação do problema** [4]. Kocev et al. [22] definem as mesmas categorias para resolução de problemas MT como métodos **globais** e **locais**, sendo que os globais se referem à adaptação do algoritmo e os locais à alteração do problema.

A abordagem global se refere à transformação de uma técnica de regressão já difundida, de forma que esta passe a lidar de forma simultânea com múltiplas saídas, além de explorar as possíveis relações entre as variáveis alvo. Esse tipo de abordagem leva a algum nível de alteração na modelagem original da técnica a ser adaptada, por exemplo, a função de otimização de SVMs [34, 35, 11], o critério de separação dos nós em árvores de regressão [10, 22], entre outros. Várias adaptações de algoritmos de regressão para problemas MT foram propostas [4], sendo aplicadas em problemas diversos [7, 36, 4]. De fato, os algoritmos pertencentes à abordagem global têm alcançado níveis satisfatórios de desempenho preditivo, trazendo a vantagem de gerar um modelo único de regressão, o que tende a tornar sua interpretação mais fácil e seu custo computacional reduzido. No entanto, a modelagem das inter-dependências entre as saídas não é uma tarefa simples, demandando mudanças conceituais profundas na formulação das técnicas. Além disso,

pelo fato de não haver a possibilidade da aplicação direta de algoritmos de regressão tradicionais, perde-se a flexibilidade em comparar o comportamento e especificidades de diferentes técnicas conhecidas de regressão em um determinado problema. Além disso, torna-se difícil explorar de forma simples as vantagens que cada uma destas técnicas poderia trazer: baixa complexidade, nível de interpretação do modelo gerado, desempenho preditivo, entre outros.

A abordagem local, na qual o método proposto nesse trabalho se situa, consiste na adaptação do problema em questão, em detrimento de se modificar um algoritmo específico. A partir da manipulação os dados de entrada, as dependências entre as variáveis de saída são exploradas e, em adição, técnicas de regressão ST são adotadas. Os modelos preditivos empregados nas soluções locais são referidos como regressores base, como já mencionado anteriormente. Apesar ter mais de um preditor para representar um problema MT, o que leva ao aumento do custo computacional de treinamento, a abordagem local traz algumas vantagens sobre a variante global. Métodos locais trazem a possibilidade de utilizar qualquer tipo de regressor base (e mesmo um conjunto híbrido de preditores), levando ao aumento do desempenho preditivo e a possível exploração de características específicas do problema a ser tratado. Assim, a abordagem local traz um aumento na modularidade e simplicidade conceitual da solução. De fato, esse tipo abordagem tem gerado desempenho preditivo aos métodos globais nos últimos anos [5, 8, 15], fator motivador para a criação de uma solução local com maior sofisticação na exploração das dependências estatísticas entre os alvos e, sua comparação com soluções locais precedentes.

3.1 Métodos globais para problemas de regressão *multi-target*

Em relação aos métodos globais, diferentes variantes de SVMs adaptadas para múltiplas saídas foram propostas [4], dentre as quais são destacados os trabalhos de Liu, Lin e Yu [34] e Xu et al. [11]. Em Liu, Lin e Yu [34], um novo modelo para cálculo de função de perda em problemas multi-objetivo é proposto, levando em consideração que uma vez que existem relações entre as saídas, estas podem ser representadas em um espaço reduzido de saída (denominado *output manifold*), onde o processo de regressão é realizado. Os resultados obtidos indicam que a técnica obtém resultados preditivos mais estáveis do que o uso de funções de perda tradicionais. No trabalho de Xu et al. [11], uma SVM para regressão MT é proposta. Segundo os autores, os modelos anteriores não levavam em consideração as relações existentes entre os *targets*, que poderiam ser, inclusive, não-lineares. Os resultados obtidos são comparáveis (mas não superiores) à abordagem ST e técnicas multivariadas, como a PLS (*Partial Least Squares Regression*, ou regressão parcial de mínimos quadrados, em português) e uma SVM multi-objetivo que não explora a dependência existente entre as variáveis de saída.

O uso de *ensembles* de árvores de decisão multi-objetivo é proposto em Kocev et

al. [10]. Os autores realizam testes utilizando *Bagging* de árvores e a técnica RF, denominada MORF (*Multi-objetive Random Forest*), sendo que a última apresentou o melhor desempenho. Para a construção dos *ensembles*, árvores PCT [37] (*Predictive Clustering Trees*, ou árvores preditivas de clusterização) foram empregadas, levando em conta que os problemas tratados contavam com várias saídas. Esse tipo de árvore utiliza uma estratégia hierárquica baseada em clusterização para particionar o conjunto de dados: o nó raiz corresponde ao *cluster* que contém todos os dados, os demais nós correspondem a outras partições nos dados através de *clusters* cada vez menores. Cada *cluster* é criado de forma a maximizar a homogeneidade dos dados que contém. Dependendo da parametrização utilizada nas PCTs, seu comportamento pode se tornar similar a algoritmos tradicionais de árvore de decisão, como a CART [37].

O método MORF é considerado um dos melhores métodos para regressão MT, considerando o desempenho preditivo, obtendo resultados similares ou superiores ao uso de abordagens ST e, sendo utilizado para comparação em outros trabalhos [9, 31, 5, 8]. No entanto, métodos locais mais recentes foram capazes de superar o desempenho preditivo da MORF [5, 8]. Adicionalmente, o uso de *Option Trees* (árvores de opção) e *Extreme Randomized Trees* (árvores extremamente aleatorizadas) foram avaliados em Osojnik, Džeroski e Kocev [14] e Kocev e Ceci [13], respectivamente, também empregando PCTs como base. Ambas as técnicas não obtiveram resultados estatisticamente superiores à MORF.

Em Aho et al. [9], um modelo baseado em um *ensemble* de regras de decisão é proposto, denominado FIRE (*Fitted Rules Ensemble*). Primeiramente, um conjunto de árvores de decisão é treinado sobre a base de dados (utilizando um procedimento similar ao da RF), do qual as regras são extraídas. Para melhorar a precisão dos resultados, modelos lineares são atribuídos a cada regra. O algoritmo FIRE então otimiza os pesos das regras, bem como os termos lineares inseridos nestas. Segundo os próprios autores do trabalho apresentado, o objetivo da técnica é oferecer um modelo que seja facilmente interpretável, mesmo que seu desempenho se situe entre os resultados de preditores menos poderosos, como o uso de uma única árvore de regressão, e a MORF.

Em Hadavandi, Shahrabi e Shamshirband [3], o uso de um *ensemble* de RNAs baseado em *Boosting*, juntamente com a extração de características com PCA é proposto. De forma simplificada, a PCA extrai novos atributos descritivos a partir dos originais, gerando novos espaços descritivos com base na variação dos dados. As novas variáveis são criadas de acordo com as direções de maior variância entre os dados de entrada, e cada nova dimensão derivada é ortogonal às anteriores. Os autores empregam um subespaço de atributos com dimensão menor que o conjunto original, composto por uma técnica de projeção derivada da PCA, para cada nova RNA adicionada ao *ensemble*. As RNAs criadas são treinadas de forma a sequencialmente minimizar o erro preditivo geral. Os

resultados obtidos pelos autores são superiores às outras técnicas *ensemble* comparadas (que são limitados a métodos que utilizam RNA). Os autores empregam RNAs do tipo MLP nos experimentos realizados.

3.2 Métodos locais para problemas de regressão *multi-target*

Nos últimos anos, algumas técnicas foram propostas para tratar problemas MT como tarefas ST, no entanto explorando as propriedades existentes entre os *targets*. Zhang et al. [35] propuseram a modificação do espaço de entrada de um problema MT através de uma técnica de virtualização dos dados. Na solução proposta pelos autores, o número de instâncias é replicado de acordo com o número de variáveis de saída e a base de treinamento passa a ter um único atributo de saída, definido como a concatenação de todas as variáveis alvo. Trata-se de uma representação com grande redundância dos dados, uma vez que os mesmos valores dos atributos serão repetidos várias vezes, apenas variando o valor de saída. Para denotar a qual das variáveis alvo a saída unificada corresponde, os autores usam uma codificação esparsa, correspondente à virtualização mencionada, onde atributos binários relacionados a cada variável de saída são adicionados ao conjunto de dados: somente o atributo correspondente ao atual alvo toma o valor lógico verdadeiro. No referido trabalho, os autores utilizaram uma máquina *Support Vector Regression* (SVR) e atingiram resultados comparáveis mas não superiores à estratégia ST.

No trabalho de Tsoumakas et al. [31], o uso de combinações lineares aleatórias (RLC - *Random Linear Combinations*) de *targets* foi proposta, de forma a explorar as relações existentes entre os atributos de saída. Na abordagem proposta, o número de variáveis alvo é aumentado através do uso de combinações lineares das variáveis dependentes, selecionadas de forma aleatória. Os coeficientes lineares são escolhidos de forma incremental e uniforme. Após a criação dos novos atributos de saída, cujo número é parametrizado, diversos problemas ST são resolvidos, um a cada novo alvo criado. Tendo obtido os resultados para cada combinação criada, um sistema linear é resolvido para obter os valores das variáveis alvo originais. Na maioria dos casos testados, a abordagem superou o desempenho obtido pelo ST e, mesmo pela MORF. No entanto, esse método considera apenas relacionamentos lineares entre os alvos, sendo que esses são selecionados de forma totalmente aleatória. A criação de um método determinístico para a exploração das dependências entre os alvos é preferível, tanto para aumentar o desempenho preditivo quanto para oferecer intuições acerca dos relacionamentos entre as saídas do problema tratado.

Na área de análise de regressão MT utilizando a adaptação do problema, alguns métodos foram adaptados de uma área intrinsecamente correlacionada: a classificação *multi-label* [4, 5]. A análise classificatória *multi-label* explora problemas que apresentam múltiplas variáveis alvo binárias, assim como em problemas de regressão MT saídas con-

tínuas são tratadas. Spyromitros-Xioufis et al. [5] propuseram duas técnicas principais baseadas na análise *multi-label*: **MTRS** (*Multi-Target Regressor Stacking*) e **ERC** (*Ensemble of Regression Chain*), respectivamente, empilhamento multi-alvo de regressores e conjunto de cadeias de regressores, em português. Nos resultados experimentais obtidos pelos autores, as técnicas obtiveram resultados similares ao método ST e favoravelmente comparáveis aos métodos MORF e RLC, dentre outros. Ambas as técnicas se baseiam na ideia de adicionar aproximações das variáveis alvo como novos atributos descritivos, inserindo, dessa forma, a influência das variáveis respostas na predição umas das outras. A adição de estimativas das variáveis de saída como atributos descritivos adicionais é referida como empilhamento de regressores.

O MTRS utiliza uma metodologia mais simples, onde a técnica empilha as predições de modelos ST para as variáveis alvo como novos atributos de entrada, juntamente com os originais, criando novos preditores treinados sobre o conjunto aumentado. Esse primeiro conjunto de preditores, cujas predições entram como atributos adicionais de entrada, são aqui referidos como uma camada de regressores. Assim, o MTRS apresenta duas camadas de regressores, a primeira, que não utiliza o empilhamento de preditores e uma segunda, que emprega as predições da primeira como atributos extras. O ERC por sua vez, se baseia na ideia de cadeias de regressores, ou *Regressor Chains* (RC). A criação de uma RC consiste em ordenar aleatoriamente as variáveis alvo e criar preditores ST para cada elemento, seguindo a ordem determinada anteriormente. Cada modelo na cadeia utiliza os atributos originais do problema juntamente com as predições dos regressores anteriores como variáveis de entrada para o treinamento. O ERC, portanto, consiste em empregar um conjunto de RCs aleatoriamente definidos, de modo a explorar diferentes combinações de influências entre os alvos. O MTRS e ERC são comparados ao método proposto e serão apresentados em detalhes em seções posteriores.

Quanto à exploração das dependências entre os alvos, o MTRS utiliza um mecanismo fixo e determinístico de empilhamento de novos atributos descritivos, o que não prevê a existência de diferentes níveis de interação entre as respostas ou mesmo a inexistência dessas relações. O ERC busca explorar as relações entre os *targets* através de diferentes ordenações das saídas, que são definidas de forma aleatória. No entanto, por se tratar de um problema de exploração combinatória, a exploração de todas as possibilidades de ordenação pode ser tornar computacionalmente proibitiva, dependendo do número de variáveis de resposta. De fato, o algoritmo do ERC prevê a utilização de dez cadeias aleatoriamente definidas quando o número de variáveis alvo ultrapassa três. Apesar de apresentarem as falhas mencionadas anteriormente, o MTRS e o ERC foram técnicas pioneiras a empregar predições dos alvos como atributos descritivos adicionais, tendo servido de base e motivação para a proposta de trabalhos posteriores [15, 8, 16, 17], dentre os quais se encontra, inclusive, uma versão inicial do método proposto, DSTARS [17].

Moyano, Gibaja e Ventura [15] propõem a utilização de algoritmos genéticos para encontrar um conjunto mais adequado de RCs, em detrimento do processo aleatório originalmente utilizado no ERC. Os autores também propuseram uma metodologia para remoção de *outliers* durante a agregação das respostas dos conjuntos de cadeias criados. O algoritmo genético foi projetado de forma a encontrar os conjuntos que minimizem o erro preditivo, o que carece o treinamento de modelos para teste, aumentando ainda mais o custo de treinamento do método em relação ao já custoso ERC. O método dos autores obteve resultados superiores ao ERC em alguns cenários, todavia não foi comparado a outras abordagens. Além disso, apenas um método de regressão base foi empregado (árvore de regressão), o que não deixa claro como o algoritmo se comporta quando combinado a outros regressores.

Melki et al. [8] propuseram o uso de uma única RC não-aleatoriamente definida. A proposta dos autores é calcular a correlação linear dentre todas variáveis alvo e, para cada uma delas, somar o total de correlação com todas as outras. A única cadeia criada é então definida iniciando com o *target* menos correlacionado, ou seja, a resposta cuja somatória de correlações foi menor, seguindo até o alvo mais correlacionado. A proposta dos autores gera ganhos em alguns cenários, mostrando que o nível de influência dos alvos pode resultar em diferenças de desempenho preditivo. No entanto, o trabalho pressupõe a existência de dependências unicamente lineares entre os alvos, o que não reflete necessariamente problemas da vida real. Além disso, a simples ordenação dos alvos não explora todos os níveis de influência que possam existir entre estes, muito menos a inexistência de relações dessa natureza. Por fim, o cálculo da influência entre os alvos (com o uso da correlação linear) utilizando somente os valores verdadeiros destes gera uma estimativa muito otimista em relação ao nível de dependência existente entre as respostas. Em um cenário ideal, as predições geradas pelos regressores para cada alvo se assemelhariam muito aos valores reais dessas grandezas, o que permitiria, sem problema algum, inserir essas estimativas como atributos extras. No entanto, em cenários reais, vários problemas tendem a surgir, como por exemplo, a presença de ruído nos dados ou a ausência de variáveis explicatórias suficientes para descrever o comportamento de um alvo. Nesses casos, por mais que uma variável alvo A possua grande influência sobre uma outra, B, de nada adiantará empregar estimativas imprecisas e errôneas da primeira para tentar descrever a segunda. De fato, o erro preditivo obtido por regressores pode fazer com que as predições obtidas pelos modelos não reflitam os relacionamentos que foram mensurados considerando apenas os valores esperados das respostas. Esse é um dos pontos explorados pelo método proposto, DSTARS, através do cálculo das dependências entre alvos comparando predições ST destes com seus repostas verdadeiras, ou valores esperados.

Recentemente, Santana, Mastelini e Barbon Jr. [16] propuseram o empilhamento de múltiplos níveis ou camadas de predições provenientes de regressores ST como atributos de entrada extras, numa abordagem de aprendizado profundo. Múltiplas estimativas das

variáveis alvo são adicionadas como atributos extras sequencialmente. As dimensões do problema são notadamente aumentadas buscando avaliar de forma exploratória e extensiva os diferentes níveis de interação entre os alvos. Os preditores sequencialmente adicionados utilizam as predições de todos os regressores precedentes como atributos extras de entrada, de forma similar ao MTRS. Similarmente ao trabalho de Melki et al. [8], a ordem de predição das variáveis de resposta segue uma ordenação, de acordo erro preditivo obtido em um conjunto de validação, da variável com menor erro preditivo para a de maior erro. Dessa forma, um alvo já predito, e que teoricamente possui valores mais fáceis de prever, entra como atributo adicional fixo, não passando mais pelo processo de empilhamento. O número de preditores ST empilhados é também determinado de forma exploratória. Apesar de representar a técnica mais custosa dentre as apresentadas e ter sido avaliada em casos pontuais, o método demonstrou gerar ganhos significativos através da exploração de múltiplos preditores empilhados.

É válido ressaltar que os métodos derivados do MTRS e ERC, apresentados anteriormente, apesar de buscarem melhorar alguns aspectos das soluções precedentes não foram capazes de obter desempenho preditivo significantemente superior a estas; tampouco consideraram e modelaram a possibilidade de existência de diferentes níveis de correlação entre as variáveis de respostas dos problemas tratados. Por essa razão, dentre os métodos locais apresentados, apenas o MTRS e ERC serão considerados para comparação com o método proposto. Métodos globais não são inseridos na análise comparativa por pertencerem a outra abordagem para resolução de problemas de regressão MT e, terem sido de forma geral, superados pelas soluções locais mais recentes.

Posteriormente às soluções descritas anteriormente, uma versão preliminar dessa pesquisa, DSTARS (*Deep Structure for Tracking Asynchronous Regressor Stacking*), foi apresentada em Mastelini et al. [17]. O algoritmo, em seu estágio inicial, empregou a ideia de encontrar o melhor número de regressores ST a serem empilhados para cada alvo. O método proposto utilizou para tal um mecanismo exploratório realizado sobre combinações de conjuntos de treino e validação, induzidos sobre o conjunto original de treinamento. O método propõe um estado de espera assíncrono para os *targets* de modo a evitar mínimos locais de erro. Nesse estado, novos regressores ST somente são criados para um alvo arbitrário quando as predições das outras respostas forem aprimoradas, permitindo diminuir o erro preditivo da saída em espera através da inserção das predições refinadas das outras respostas como atributos extras. Em outras palavras, regressores não são criados para certos alvos em algumas iterações da solução proposta se as predições disponíveis para as outras respostas não forem suficientemente acuradas. Todavia, apesar de apresentar um mecanismo sofisticado para explorar os diferentes níveis de influência entre os alvos e ter apresentado melhorias em relação ao MTRS e ERC, o DSTARS em sua forma inicial não mensurava explicitamente o nível de interação entre as variáveis alvo para posterior submissão ao seu processo de empilhamento de camadas de regressores ST.

O método proposto, assim como sua forma inicial apresentada em Mastelini et al. [17], aplica a ideia proposta no MTRS e ERC de utilizar estimativas ST das variáveis de saída como informações adicionais para descrevê-las. A hipótese para o DSTARS é que se uma segunda camada de preditores ST, ou um segundo conjuntos destes, pode oferecer um melhor desempenho preditivo (assim como é proposto no MTRS), aproximações advindas de camadas mais profundas de regressores podem gerar erros de predição ainda menores, dependendo do nível de interação das variáveis alvo. Como já mencionado, a técnica proposta utiliza a composição assíncrona de camadas de preditores, avaliando cada variável alvo de forma individual, de acordo com o erro obtido em conjuntos de validação. Diferente dos métodos locais existentes até então, o DSTARS considera as peculiaridades de cada variável alvo, atribuindo um número diferente de regressores empilhados para cada uma. De forma sequencial, os atributos extras adicionados, que correspondem à predições das variáveis alvo, são substituídos por aproximações cada vez mais refinadas, levando à diminuição do erro preditivo. Esse aspecto, juntamente com o estado de espera previsto no DSTARS, e mencionado anteriormente, diferencia o método proposto do MTRS em alguns pontos importantes. Primeiramente, novamente destacando, o DSTARS empilha um número diferente e dinâmico de regressores para cada alvo, enquanto que o MTRS sempre empilha um regressor para cada alvo. Dessa forma, no DSTARS, um problema pode gerar uma pilha profunda de modelos regressores para uma saída específica e predições ST para outro alvo, dependendo da correlação existente entre os dados. Assim, alvos com maiores dependências em relação aos outros gerarão um maior número de regressores empilhados, enquanto os menos dependentes originarão um menor número de preditores empilhados. Segundo, em uma dada camada de regressores, o preditor para um alvo específico pode não estar presente, decorrente do estado de espera mencionado, enquanto que nas camadas de regressores induzidas pelo MTRS, preditores para todos os alvos sempre estão presentes.

Em adição à versão preliminarmente apresentado em Mastelini et al. [17], um mecanismo explícito e realista de descrição das relações entre as variáveis alvo é proposto para o DSTARS, de modo a filtrar e submeter ao processo de empilhamento de preditores ST somente as variáveis alvo que possuam dependências entre si. Para tal, uma métrica não-linear de importância de variáveis é empregada para a descrição dos relacionamentos entre os alvos, sendo tal grandeza derivada do algoritmo RF [25]. Todavia, diferentemente de soluções anteriores na área [15, 8], ao invés de utilizar os valores verdadeiros dos alvos para o cálculo das influências que estes exercem uns nos outros, o DSTARS avalia como estimativas (predições ST) das variáveis resposta, avaliadas em conjuntos de validação, se relacionam aos valores observados ou esperados destas.

Preliminarmente à apresentação do DSTARS, as próximas seções apresentam em detalhes o MTRS e o ERC. Discussões acerca da complexidade computacional desses dois métodos, bem como seus algoritmos são apresentados, tendo em vista que essas soluções

locais são comparadas, juntamente com a estratégia ST, ao método proposto.

3.2.1 MTRS: *Multi-target Regressor Stacking*

O método MTRS consiste em separadamente treinar modelos ST e utilizar as saídas dos regressores induzidos como atributos preditivos adicionais para mais uma camada de regressores ST. Dessa forma, levando em consideração um conjunto de entrada X, composto por m variáveis na forma $X = \{x_1, x_2, ..., x_m\}$, e também d variáveis alvo Y, na forma $Y = \{y_1, y_2, ..., y_d\}$, o MTRS usa as predições ST $\overset{p}{Y} = \{\overset{p}{y_1}, \overset{p}{y_2}, ..., \overset{p}{y_d}\}$ advindas de modelos treinados sobre X como entradas adicionais, formando um novo conjunto de entrada estendido $\overset{est}{X} = \{x_1, x_2, ..., x_m, \overset{p}{y_1}, \overset{p}{y_2}, ..., \overset{p}{y_d}\}$. Esse conjunto transformado é utilizado juntamente com os valores de Y para treinar uma camada adicional de preditores ST, cujas saídas são as predições finais. A Figura 2 apresenta uma representação gráfica do procedimento do MTRS. Na Figura, X representa os atributos de entrada, $\overset{p}{y_i}$ e $\overset{F}{y_i}$ com $i \in \{1, 2, ..., d\}$, representam, respectivamente, as predições das variáveis alvo para a primeira e a segunda camada de regressores ST empilhados.

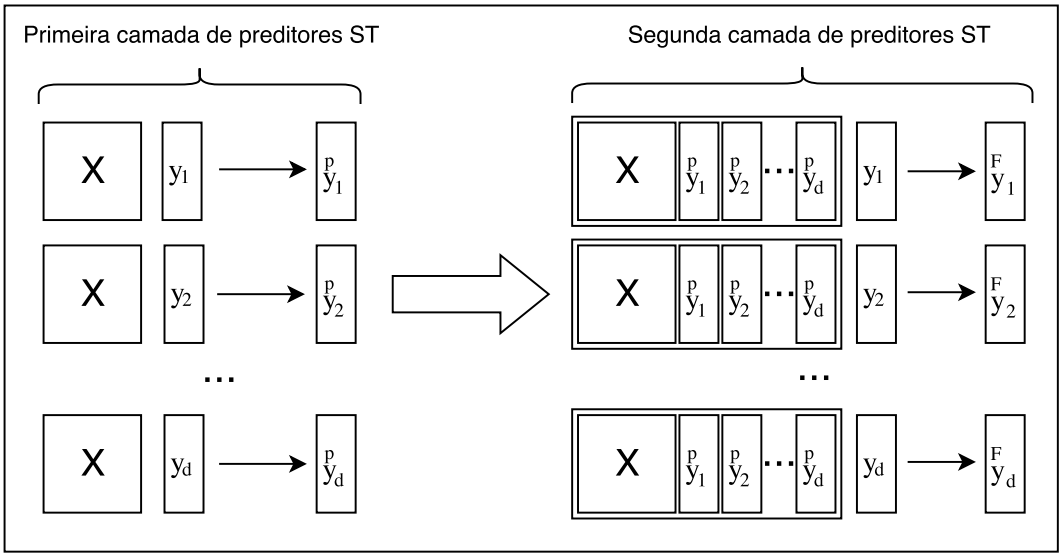

Figura 2 – Representação gráfica do procedimento de treinamento do MTRS

Quando novos dados são submetidos para análise, as variáveis de entrada são primeiramente submetidas à primeira camada de preditores para a obtenção das aproximações das saídas e, assim, compor um conjunto estendido de teste que é sujeito ao segundo nível de regressores para originar as estimativas finais. Apesar de utilizar apenas modelos ST, a dependência estatística entre os *targets* pode ser explorada e inserida na modelagem, consequentemente aumentando a capacidade de descrição de problemas MT e o desempenho obtido. As correlações entre os alvos são exploradas a partir do momento que estimativas destes são empregadas como atributos de entrada, permitindo utilizar as inter-dependências como fonte adicional de informação. No entanto, como já ressal-

tado anteriormente, estimativas de todos as saídas serão empilhadas, independente de sua correlação com o alvo para o qual um modelo de regressão esteja sendo construído.

O procedimento de treinamento no MTRS é apresentado no Algoritmo 1. Os parâmetros para o algoritmo são os atributos de entrada, as variáveis de saída e o número de alvos, representados, respectivamente, por X, Y e d.

Algoritmo 1 Algoritmo de treinamento do MTRS

Função MTRS(X, Y, d)
    // Para armazenar as predições ST
    $\overset{p}{Y} \leftarrow \{\}$
    // Primeira camada de modelos ST
    $\text{Nível}_0 \leftarrow \{\}$
    // Indução de regressor ST para cada alvo
    para t $\leftarrow 1$ **até** d **faça**
        // Indução de regressor ST
        $h : X \rightarrow Y_t$
        $\overset{p}{Y}_t \leftarrow \text{predição}(h, X)$
        $\text{Nível}_0 \leftarrow \text{Nível}_0 \cup h$
    fim para
    // Criação de conjunto de treinamento expandido
    $\overset{est}{X} \leftarrow X || \overset{p}{Y}$
    // Segunda camada de modelos ST
    $\text{Nível}_1 \leftarrow \{\}$
    para t $\leftarrow 1$ **até** d **faça**
        $h : \overset{est}{X} \rightarrow Y_t$
        $\text{Nível}_1 \leftarrow \text{Nível}_1 \cup h$
    fim para
    $mtrs \leftarrow \text{Nível}_0 \cup \text{Nível}_1$
    retorna $mtrs$
fim Função

Em relação ao custo computacional em relação ao tempo de treinamento, considerando b como a complexidade do regressor base adotado, a complexidade assimtótica do MTRS é $O(2\,d\,b)$. Essa complexidade representa o treinamento do dobro de modelos que a abordagem ST, cuja complexidade de tempo, por consequência, é $O(d\,b)$. É válido ressaltar que o termo N, relativo ao número de instâncias do problema tratado, não aparece nas expressões de cálculo de complexidade pelo fato dessa grandeza estar embutida no custo de treinamento do regressor escolhido (b).

3.2.2 ERC: *Ensemble of Regressor Chains*

Também proposto em Spyromitros-Xioufis et al. [5], o ERC consiste em utilizar um conjunto aleatoriamente selecionado e ordenado de sequências de variáveis de saída para construir modelos ST encadeados, seguindo a ordem determinada por essas cadeias,

as RCs. Como já mencionado anteriormente, para cada RC, inicialmente, um regressor ST é induzido utilizando como alvo a primeira variável de saída da sequência. Novos modelos são treinados seguindo a ordenação da RC, sendo que cada novo preditor utiliza um conjunto de treinamento estendido, formado pela combinação dos atributos originais do problema e as predições dos modelos anteriores, pertencentes à mesma cadeia. O processo é repetido até o fim da sequência definida, e para cada RC determinada.

Novas instâncias são submetidas a todos os regressores, de forma sequencial, assim como acontece no MTRS. A predição final para uma variável de saída específica, y_i, é a média tomada dentre as respostas de todos os modelos do *ensemble* que geram estimativas de y_i, sendo que $i \in \{1, 2, ..., d\}$. Levando em consideração que as estimativas geradas pelo ERC para cada saída são compostas por valores originados em diferentes posições das cadeias de regressores, múltiplos níveis de combinações e inter-dependências entre *targets* são explorados, ainda que de forma exaustiva. No entanto, pelo número potencialmente grande de permutações possíveis para as saídas, na formulação original do ERC apenas dez cadeias aleatórias são geradas se o número de variáveis alvo é maior do que três. Caso contrário, todas as ordenações possíveis são selecionadas.

Algoritmo 2 Algoritmo de treinamento do ERC

Função ERC(X, Y, d)
  targets $\leftarrow$ nomes(Y)
  se $d \leq 3$ **então**
    Chains $\leftarrow$ permutar(targets)
  senão
    Chains $\leftarrow$ permutar(targets, 10)
  fim se
  para cada chain $\in$ Chains **faça**
    $\text{regressores}_{\text{chain}} \leftarrow \{\}$
    // Para construir conjuntos de treinamento estendidos
    $\overset{est}{X} \leftarrow X$
    para cada t $\in$ chain **faça**
      // Indução de regressor ST
      $h : \overset{est}{X} \rightarrow Y_t$
      $y_{\text{pred}} \leftarrow$ predição($h, \overset{est}{X}$)
      // Adiciona predição como atributo descritivo extra
      $\overset{est}{X} \leftarrow \overset{est}{X} || y_{\text{pred}}$
      $\text{regressores}_{\text{chain}} \leftarrow \text{regressores}_{\text{chain}} \cup h$
    fim para
    $erc \leftarrow erc \cup \text{regressores}_{\text{chain}}$
  fim para
  retorna erc
fim Função

O procedimento de treinamento para o ERC é apresentado no Algoritmo 2. Neste,

os parâmetros de entrada são as variáveis de entrada (X), os alvos (Y) e o número de alvos (d). Além disso, a função *permutar*, invocada dentro do pseudocódigo do ERC, representa um procedimento em que dado um conjunto de entrada: (1) retorna todas permutações possíveis entre seus elementos, ou (2) retorna exatamente o número de permutações passado como parâmetro adicional, sendo essas permutações selecionadas aleatoriamente.

Em relação à complexidade de tempo do ERC, levando em consideração b como a complexidade do regressor base escolhido, temos duas possibilidades. O primeiro caso corresponde a levar em consideração todas as permutações possíveis entre as variáveis alvo. Dessa forma, a complexidade do ERC é $O(d^2\,(d-1)!\,b)$, uma vez que para cada cadeia gerada d regressores serão induzidos. No entanto, ao seguir a recomendação de Spyromitros-Xioufis et al. [5] de utilizar apenas dez cadeias (aleatoriamente definidas) se o número de alvo ultrapassar três, a complexidade obtida para o ERC é $O(10\,d\,b)$.

Com o intuito de melhor visualização dos procedimentos realizados pelo ERC, a Figura 3 apresenta o procedimento de indução de modelos ST para uma RC, o que corresponde a uma dentre as possíveis permutações para as variáveis alvo. No caso, como forma de exemplificação, a ordenação natural das variáveis resposta está representada, partindo de y_1 até y_d. Os valores de $\overset{p}{y}_i$, com $i \in \{1, 2, ..., d\}$, representam as saídas dos modelos ST treinados na referida cadeia de alvos. O algoritmo do ERC induz múltiplas cadeias como a apresentada na figura, agregando as respostas destas com a operação de média, para cada predição $\overset{p}{y}_i$.

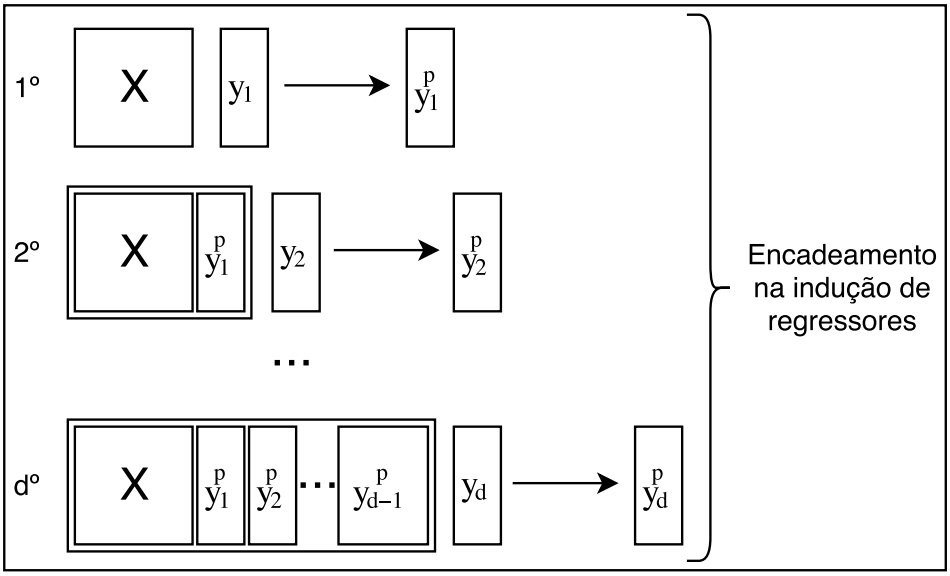

Figura 3 – Representação gráfica do procedimento de treinamento de uma RC

4 DSTARS: *DEEP STRUCTURE FOR TRACKING ASYNCHRONOUS REGRESSOR STACKING*

O método proposto, DSTARS, está pautado na premissa de que a indução de camadas potencialmente profundas de modelos de regressão empilhados deve originar um desempenho preditivo superior, quando comparado a métodos mais rasos, como o uso de uma única camada (ST) ou duas (MTRS). Dessa forma, ao invés de utilizar atributos extras originados de uma única camada de estimadores, assim como realizado na técnica MTRS, o método DSTARS determina qual a melhor disposição de camadas de regressores para cada *target*, a partir da sucessiva construção e avaliação de preditores para as saídas do problema, buscando minimizar o erro preditivo em conjuntos de validação. Dessa forma, os modelos empilhados são refinados gradualmente, utilizando predições cada vez mais acuradas das saídas como atributos extras, além de prover melhores predições para os regressores dos outros *targets*, também sujeitos ao empilhamento.

Além das características já mencionadas, o DSTARS também parte do pressuposto que existem casos em que nem todas as múltiplas variáveis de um problema de regressão MT possuem dependências estatísticas entre si. Para tal, uma métrica não-linear de importância de variáveis é empregada para avaliar e descrever como as variáveis alvo se relacionam umas com as outras. Assim, estarão sujeitas ao empilhamento de regressores somente os *targets* que possuam dependências estatísticas em relação às outras respostas. Um alvo não-dependente será então tratado como um problema ST individual.

A Figura 4 apresenta uma representação esquemática do método proposto. O DSTARS se inicia a partir da separação dos dados de entrada em conjuntos de treino e validação, utilizando para tal uma abordagem de amostragem, como a validação cruzada. Na figura, f combinações de treino e validação são representadas. Essa separação se faz necessária para a busca do número mais adequado de camadas de regressores para cada alvo. Utilizando as partições de treino, uma primeira camada de modelos ST é antecipadamente induzida, uma vez que suas predições serão relevantes para a mensuração das correlações existentes entre as variáveis de resposta. As predições resultantes dessa primeira camada de preditores estão assinaladas com o sobrescrito 1 na Figura 4. Após a definição desses conjuntos e da primeira camada de regressores ST, o DSTARS é dividido em três etapas principais, indicadas na Figura 4: *Filtering*, *Tracking* e *Modelling*.

A etapa de *Filtering* determina quais variáveis alvo possuem influências e são influenciadas pelas outras, sendo que somente as variáveis de saída correlacionadas são escolhidas para o empilhamento de regressores ST. Como indicado na Figura 4, as predições avaliadas nos conjuntos de validação, obtidas anteriormente, são comparadas com os valores verdadeiros das variáveis alvo. Uma métrica de importância de variáveis é em-

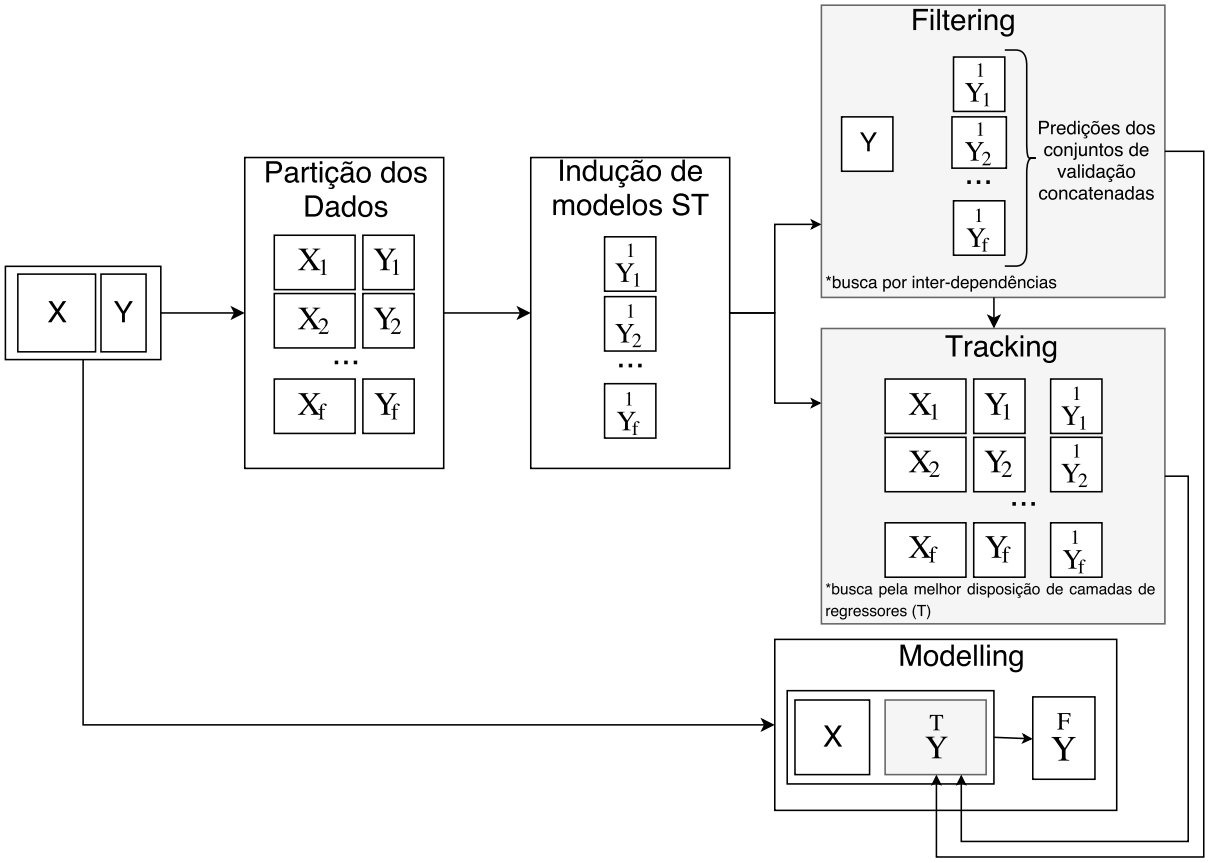

Figura 4 – Representação esquemática do método proposto

pregada para avaliar como os dois conjuntos mencionados se relacionam. Utilizando essa informação, a etapa de *Tracking* busca determinar a melhor disposição de camadas de regressores para o problema tratado, explorando suas características nos conjuntos de treino e validação. As porções separadas para treino são utilizadas para sucessivamente adicionar novos regressores empilhados, correspondentes aos alvos que possuem dependências estatísticas. Os modelos treinados são avaliados nas partições de validação, verificando se estes foram capazes de reduzir o erro preditivo existente. Em caso de redução de erro, as saídas de tais modelos serão utilizadas como atributos extras na próxima iteração; caso contrário, os melhores resultados obtidos até então para aquele alvo serão utilizados.

O processo iterativo para quando não houver ganhos significativos de desempenho para nenhuma das variáveis de resposta. Durante o processo descrito, cada combinação de alvo/camada que originou uma diminuição no erro preditivo (calculado nos conjuntos de validação) é registrada. Na existência de múltiplas repetições na amostragem de conjuntos de treino e validação, em um procedimento de validação cruzada por exemplo, todas as camadas utilizadas para cada *target* passam por um processo de voto para a escolha das quais possuem maior relevância. Vale mencionar que os regressores treinados nesse processo exploratório não são armazenados, uma vez que não correspondem aos modelos preditivos definitivos e sim, meios para analisar as características do problema tratado.

Como último passo, a etapa de *Modelling* utiliza todos os dados descritivos disponíveis para criar regressores seguindo a estrutura determinada na etapa anterior, indicada por T na Figura 4. Os regressores treinados nessa fase resultam no conjunto final de modelos preditivos do DSTARS, cujas saídas estão indicadas com o sobrescrito F na mesma figura.

A seguir, primeiramente, serão introduzidas as convenções de notação utilizadas para a apresentação do método proposto. Em seguida, o funcionamento geral do DSTARS e cada uma de suas etapas serão apresentadas em detalhes.

4.1 Convenções de notação utilizadas

Nas próximas seções, informações detalhadas acerca do DSTARS serão apresentadas. Para tal, algumas convenções para notação foram adotadas e serão introduzidas a seguir.

Um problema de regressão MT corresponde ao relacionamento de um conjunto de m variáveis de entrada, X, a d variáveis resposta contínuas, Y. A notação *sub-escrito* indica o acesso a uma única variável ou atributo, como por exemplo, o acesso à i-ésima variável de X por X_i. Em adição, a notação *sobre-escrito* indica qual camada de regressores está sendo manipulada, como por exemplo, acessar a l-ésima camada da t-ésima variável de saída em um modelo DSTARS treinado, através de dstars_t^l.

Identificadores acima de nomes de variáveis tem o propósito de diferenciá-las, como em $\overset{tr}{X}$ e $\overset{va}{X}$, que representam os atributos de entrada para os conjuntos de treino e validação, respectivamente. A notação $||$ indica a concatenação de dois conjuntos que tenham o mesmo número de instâncias.

A função `nLinhas` retorna o número de linhas em uma matriz, enquanto a função `concatenaLinhas` retorna a concatenação das linhas de duas matrizes que possuam o mesmo número de colunas. Por fim, o acesso a matrizes e vetores é definido através de colchetes, enquanto que a manipulação de conjuntos é definida pelo uso de chaves. A criação de uma matriz de tamanho arbitrário é realizada através da indicação das dimensões da estrutura desejada dentro de colchetes, seguida pela atribuição de um valor inicial que será compartilhado por todos os seus elementos internos. Em adição, a criação de um conjunto vazio é dada por $\{\}$.

4.2 Visão geral do DSTARS

Como anteriormente mencionado, o método DSTARS utiliza informações de modelos treinados e testados em subpartições dos dados originais apresentados para treinamento. Os erros obtidos nos conjuntos de validação são utilizados tanto para mensurar o nível das dependências estatísticas entre os alvos quanto para determinar a melhor dis-

posição de camadas de regressores. O Algoritmo 3 apresenta uma visão geral de todos os passos realizados pelo DSTARS. Os parâmetros passados para o método são X e Y, que representam, respectivamente, os conjuntos de entrada e saída para o aprendizado, ϕ, que corresponde ao limiar para escolha de camadas utilizado pelo *Tracking*, ε, que é o critério de convergência de erro para limitar a adição de novos regressores, e `tipo_amostragem`, que indica a estratégia de particionamento dos dados empregada (validação cruzada, *hold-dout*, *bootstrap*, etc.).

Algoritmo 3 Procedimento completo do DSTARS

1: **Função** DSTARS(X, Y, ϕ, ε, tipo_amostragem)
2: Partições $\leftarrow$ Particionar($X, Y,$ tipo_amostragem)
3: $part \leftarrow 1$
4: // Armazenamento das predições ST para a etapa de *Filtering*
5: $\overset{st}{Y} \leftarrow \{\}$
6: // Indução de modelos ST nas partições para as etapas de *Filtering* e *Tracking*
7: **para cada** $\{(\overset{tr}{X}, \overset{tr}{Y}), (\overset{va}{X}, \overset{va}{Y})\} \in$ Partições **faça**
8: // Armazenamento dos erros obtidos em cada partição
9: $erros_{1..d} \leftarrow \infty$
10: // Variáveis para armazenamento das predições obtidas nas partições
11: $\overset{ptr}{Y}[nLinhas(\overset{tr}{Y}), d] \leftarrow 0$
12: $\overset{pva}{Y}[nLinhas(\overset{va}{Y}), d] \leftarrow 0$
13: // Indução de modelos ST
14: **para** t $\leftarrow 1$ **até** d **faça**
15: $mod : \overset{tr}{X} \rightarrow \overset{tr}{Y_t}$
16: $\overset{ptr}{Y_t} \leftarrow$ predição($mod, \overset{tr}{X}$)
17: $\overset{pva}{Y_t} \leftarrow$ predição($mod, \overset{va}{X}$)
18: $erros_t \leftarrow$ RMSE($\overset{va}{Y_t}, \overset{pva}{Y_t}$)
19: **fim para**
20: Partições$_{part} \leftarrow$ Partições$_{part} \cup \{\overset{ptr}{Y}, \overset{pva}{Y}, erros\}$
21: $\overset{st}{Y} \leftarrow$ concatenaLinhas($\overset{st}{Y}, \overset{pva}{Y}$)
22: $part \leftarrow part + 1$
23: **fim para**
24: $F \leftarrow$ Filtering($\overset{st}{Y}, Y$)
25: $T \leftarrow$ Tracking(Partições, ϕ, ε, F)
26: $dstars \leftarrow$ Modelling(X, Y, T, F)
27: **retorna** $dstars$
28: **fim Função**

Como pode ser observado no algoritmo apresentado, inicialmente os conjuntos X e Y são particionados utilizando a estratégia definida em tipo_amostragem (linha 2). Para cada uma das partições de treino e validação, representadas pelas anotações tr e va acima das variáveis correspondentes, modelos de regressão ST são treinados. Essa ação corresponde à primeira camada de regressores, que sempre estará presente para

todos os problemas tratados pelo DSTARS. Tal camada corresponde aos modelos que não empregam conjuntos aumentados por estimativas dos *targets*, utilizando, portanto, apenas os atributos descritivos originais do problema tratado. Como será descrito com maiores detalhes na seção correspondente à etapa de *Tracking*, para as camadas posteriores regressores não serão necessariamente treinados para todos os alvos, implicando no termo *assíncrono* presente no nome do método proposto.

As predições obtidas pelos modelos ST da primeira camada, nos conjuntos de treino e validação, são armazenadas tendo em vista que poderão ser empregadas como atributos extras das próximas e camadas. Em adição, os erros preditivos obtidos nos conjuntos de validação são calculados (linha 18), sendo que serão empregados para mensurar os ganhos obtidos pela adição de camadas adicionais de regressores para cada alvo. A métrica escolhida para o cálculo do erro foi a raiz do erro quadrático, ou *Root Mean Squared Error* (RMSE), como indicado na Equação 4.1.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4.1}$$

Na equação, n representa o número de instâncias no conjunto avaliado, y_i os valores reais na i-ésima instância do alvo analisado e $\hat{y}_i$ as predições obtidas para a mesma instância por um modelo de regressão. Como mostrado na linha 20 do Algoritmo 3, as predições e os erros obtidos nas partições criadas são armazenados juntamente com os conjuntos particionados para serem utilizados por etapas posteriores do DSTARS. Por fim, as predições nos conjuntos de validação são unificadas em um único conjunto $\overset{st}{Y}$ (linha 21 do Algoritmo 3) que é empregado na etapa de *Filtering*.

4.3 Filtering

A primeira etapa principal do DSTARS consiste na determinação do nível de influência que cada variável de saída tem sobre as outras. Dessa forma, somente as variáveis com inter-dependências são submetidas ao empilhamento de regressores. Trabalhos prévios utilizaram a correlação linear entre as saídas como indicativos das influências existentes entre as variáveis resposta [8, 15], no entanto, não é seguro afirmar que somente relações lineares poderão existir entre os *targets*. Por essa razão, o emprego de uma métrica capaz de capturar relacionamentos não-lineares entre os alvos será capaz de melhor modelar as características intrínsecas aos problemas tratados.

Assim, levando em consideração que os problemas tratados envolvem regressão, a métrica de importância de variáveis da RF foi empregada para esse fim. Essa métrica, aqui referida como `RFimp`, tem sua origem nas próprias características do *ensemble* que a origina. Como já mencionado na Seção 2.1.3, cada preditor adicionado à floresta em-

prega uma subamostragem das variáveis de entrada e uma base de dados de treinamento diferente, que é formada a partir do conjunto de dados original e possui o mesmo tamanho deste. Tais conjuntos de treino são compostos a partir da amostragem aleatória com reposição das instâncias, o que gera a repetição na escolha de certas amostras enquanto outras nunca são selecionadas. Assim, cada preditor na floresta possui um subconjunto de instâncias que não foram utilizadas em seu treinamento, sendo chamadas de amostras *Out-of-Bag* (OOB) [25]. Essas instâncias podem ser utilizadas para avaliar cada árvore individualmente, sem um viés de erro, e são empregadas para avaliar o impacto que cada variável de entrada do problema possui no erro preditivo obtido pela floresta. O erro preditivo calculado sobre todas as amostras OOB é referenciado como erro OOB, ou *Out-of-Bag Error* (OOBE). Para tal, o erro quadrático médio, ou *Mean Squared Error* (MSE) é empregado, assim como proposto pelo autor do algoritmo da RF [25]. O cálculo do MSE é apresentado na Equação 4.2, na qual N representa o número de instâncias avaliadas e y e $\hat{y}$ indicam, respectivamente, os valores verdadeiros dos alvos avaliados e as predições obtidas por um regressor.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{4.2}$$

Para cada variável de entrada de um problema de regressão, a métrica `RFimp` é calculada da seguinte forma, considerando um modelo RF já treinado:

1. O OOBE é calculado e registrado;
2. Os valores da variável de entrada em questão são aleatoriamente permutados entre as instâncias OOB;
3. As amostras OOB alteradas são novamente submetidas à floresta para predição e o OOBE obtido é registrado;
4. O aumento no OOBE (porcentagem de aumento) resultante após a permutação da variável de entrada passa a corresponder ao seu valor de `RFimp`.

Dessa forma, quanto maior o aumento no erro resultante da permutação nos valores de uma variável, maior a importância desta para o problema tratado. No DSTARS, a `RFimp` é calculada para todas as variáveis alvo. Como essa métrica não pressupõe um relacionamento linear entre as variáveis de entrada e a saída correspondente, pode ser empregada como um mensurador não-linear de dependência entre os *targets*. Apesar da `RFimp` ter sido escolhida para descrever a relação entre os *targets*, qualquer outra métrica de importância pode ser aplicada, como por exemplo, a correlação linear entre os alvos.

Essa escolha fica a critério do utilizador. Todavia, devido ao interesse em mensurar correlações não-lineares, a `RFimp` foi empregada nos experimentos em detrimento de outras métricas.

No entanto, o emprego dos valores verdadeiros dos alvos para a mensuração das correlações existentes pressupõe um cenário muito otimista. Na prática, o valor de erro para um alvo em um problema de regressão pode ser muito alto e dessa forma, as predições obtidas por preditor podem não apresentar as dependências estatísticas existentes e mensuradas a partir dos valores esperados para as respostas. Por essa razão, o DSTARS realiza o cálculo da `RFimp` sobre as predições da primeira camada de modelos ST que foram anteriormente treinados e avaliados nos conjuntos de validação. Após o cálculo das importâncias, somente os atributos que possuem valores de `RFimp` maiores do que zero serão submetidos ao empilhamento de regressores.

O Algoritmo 4 apresenta o procedimento de *Filtering* do DSTARS. Neste pseudocódigo, a função de filtragem recebe como parâmetros duas matrizes numéricas $\overset{st}{Y}$ e Y de mesmas dimensões, correspondentes às predições dos modelos ST treinados sobre os conjuntos de validação e os valores verdadeiros observados para essas instâncias. No procedimento apresentado, a função `RFimp` treina um modelo RF e utiliza este para calcular a importância das variáveis em $\overset{st}{Y}$ para cada um dos atributos de saída em Y.

Algoritmo 4 Algoritmo para etapa de *Filtering* do DSTARS

```
1: Função FILTERING(Ŷ^st, Y)
2:     // Criação de matriz para armazenar os valores de importância calculados
3:     F[d, d] ← Falso
4:     para t ← 1 até d faça
5:         aux ← RFimp(Ŷ^st, Y_t)
6:         // Seleciona os atributos de Ŷ^st com importância positiva em Y_t
7:         para u ← 1 até d faça
8:             se aux[u] > 0 então
9:                 F[t, u] ← Verdadeiro
10:            fim se
11:        fim para
12:    fim para
13: retorna F
14: fim Função
```

4.4 Tracking

Após a definição das dependências estatísticas existentes entre as variáveis resposta, a próxima etapa para o DSTARS é a definição da disposição mais adequada de camadas de regressores para cada *target*. Essa ação é feita com base nas partições de treino e validação definidas nas etapas anteriores. A etapa de *Tracking* é apresentada no

Algoritmo 5. Os parâmetros passados para essa etapa são as partições de treino e validação (juntamente com as predições e erros dos modelos ST, definidos na Seção 4.2), o limiar ϕ, o critério de convergência de erro ε e os valores de filtragem F, definidos na etapa de *Filtering*.

A ideia básica por trás desse procedimento é adicionar novos regressores para cada alvo enquanto houver ganhos preditivos. Cada novo regressor utiliza atributos explanatórios extras, correspondentes às predições dos modelos que obtiveram os menores índices de erro nos conjuntos de validação. Cada vez que ganhos são obtidos, a combinação *target*/camada que os gerou é registrada. Assim, uma trilha das melhores camadas por alvo é guardada. Em um cenário com múltiplas partições de treino e validação, como na validação cruzada, cada vez que uma camada for empregada para um *target*, esta receberá um voto (linha 23, Algoritmo 5), correspondente ao registro de utilização já mencionado.

No treinamento de novos regressores para um *target*, as variáveis de resposta que o influenciam são selecionadas (linha 12) e suas predições empregadas como atributos descritivos extras para o problema. Cada vez que um novo regressor é treinado para um alvo específico t (linha 14), o erro preditivo obtido para essa saída é avaliado na partição de validação (linha 17). Esse erro é então comparado com o menor erro de predição de t obtido até o momento, na partição de validação avaliada. Caso seja observada uma diminuição maior que ε no erro (linha 19), assume-se que o alvo em questão não atingiu um estado de convergência (linha 20) e continuará a ser explorado. Nesse caso, o erro preditivo obtido para t passa a ser a nova melhor aproximação para esse alvo (linha 21), a camada em questão ganha um voto para t (linha 23) e as predições de t nos conjuntos de treino e validação obtidas na camada atual passam a ser empregadas como atributos extras para as próximas camadas (linhas 24 e 25). Caso o erro em t diminua em um valor inferior a ε, ou aumente, a variável em questão assume um estado de convergência (linha 27).

É importante ressaltar que mesmo que um *target* assuma o estado de convergência, enquanto esse fato não se estender para todas as outras respostas, o alvo em questão fica em um estado de espera. Em outras palavras, modelos preditivos continuarão a serem treinados para t nas próximas iterações, empregando as melhores aproximações existentes para as saídas como atributos adicionais. Eventualmente um novo modelo pode trazer ganhos suficientes para t, devido a aproximações mais refinadas dos alvos, fazendo o *target* em questão sair do estado de convergência. Nesse cenário, a trilha de camadas de um alvo pode conter descontinuidades, correspondentes às iterações do *Tracking* em que este estava em um estado de espera. Tais descontinuidades motivam a característica de assincronismo indicada no nome do método proposto, DSTARS.

Como exemplificação, considerando d variáveis de saída, a Tabela 1 apresenta a utilização das camadas de regressores em um cenário hipotético para um *target* específico

Algoritmo 5 Algoritmo para etapa de *Tracking* do DSTARS

Função TRACKING(Partições, ϕ, ε, F)
  // Contagem dinâmica da utilização de camadas de regressores
  $T[?, d] \leftarrow 0$
  para cada $\{(\overset{tr}{X}, \overset{tr}{Y}), (\overset{va}{X}, \overset{va}{Y}), (\overset{ptr}{Y}, \overset{pva}{Y}), erros\} \in$ Partições **faça**
    // Em todos os casos a primeira camada (sem atributos extras) é utilizada
    $T[1, 1..d] \leftarrow T[1, 1..d] + 1$
    camada $\leftarrow 2$
    convergiram$_{1..d} \leftarrow$ **Falso**
    enquanto !*todos*(convergiram) **faça**
      para t $\leftarrow 1$ **até** d **faça**
        // Filtragem dos *targets* relevantes para t
        $f \leftarrow F[t, 1..d]$
        // Indução de regressor ST em conjunto de treinamento aumentado
        $mod : \overset{tr}{X} || \overset{ptr}{Y_f} \rightarrow \overset{tr}{Y_t}$
        $prd \leftarrow$ predição$(mod, \overset{va}{X} || \overset{pva}{Y_f})$
        // Erro preditivo no conjunto de validação
        $ep \leftarrow \text{RMSE}(\overset{va}{Y_t}, prd)$
        // Critério de parada para o empilhamento
        se $ep + \varepsilon <$ errors$_t$ **então**
          convergiram$_t \leftarrow$ **Falso**
          errors$_t \leftarrow ep$
          // Marca que camada trouxe melhorias de erro
          $T[\text{camada}, t] \leftarrow T[\text{camada}, t] + 1$
          $\overset{ptr}{Y_t} \leftarrow$ predição$(mod, \overset{tr}{X} || \overset{ptr}{Y_f})$
          $\overset{pva}{Y_t} \leftarrow prd$
        senão
          convergiram$_t \leftarrow$ **Verdadeiro**
        fim se
      fim para
      camada $\leftarrow$ camada +1
    fim enquanto
  fim para
  $nPart \leftarrow$ quantidade(Partições)
  $T \leftarrow T/nPart$ // Normalização da utilização das camadas
  // Aplicação do limiar: valores **Verdadeiro** e **Falso**
  $T \leftarrow T > \phi$
retorna T
fim Função

t. Por simplicidade, o uso de camadas para os outros $d - 1$ alvos não é apresentado, todavia, em cada uma das iterações supõe-se que regressores são induzidos para todos as respostas, utilizando em todos os casos aproximações das variáveis de saída que possuam influências estatísticas como atributos adicionais de entrada.

Neste cenário, a segunda camada de regressores para o alvo em questão utilizou as predições de t da primeira (camada ST) como atributo extra e alcançou uma diminuição

Tabela 1 – Exemplo hipotético de utilização de camadas para um *target* durante o *Tracking*

Camada atual	Camada empilhada	Ganho
2	1	Sim
3	2	Não
4	2	Não
5	2	Sim
6	5	Sim

relevante (superior a ε) no erro preditivo sobre o conjunto de validação. A camada 3 não obteve êxito em diminuir o erro preditivo para o alvo tratado. Similarmente, a camada 4 utilizando as predições da última camada que trouxe melhorias (camada 2) não diminuiu os índices de erro obtido. No entanto, as predições obtidas para os outros alvos, não representados nesse exemplo, foram refinadas nesse meio tempo. Por essa razão, a camada 5, empregando os valores de t obtidos na camada 2 (e as predições dos outros alvos não apresentados aqui) foi capaz de reduzir em mais de ε o erro preditivo obtido, passando a ser a melhor aproximação vigente de t. Assim, as predições da camada 5 foram empregadas na camada 6 para a indução de novos regressores.

Após o fim do processo iterativo de busca pelas camadas de regressores relevantes ao problema, uma etapa de limiarização é realizada para escolher as camadas que serão utilizadas no modelo final DSTARS. Esse procedimento é apresentado nas últimas linhas do Algoritmo 5. Primeiramente, a contagem de utilização de camadas por alvo é normalizada pelo número de partições utilizadas (linha 34). Dessa forma, os valores de contagem passam a variar no intervalo $[0, 1]$, refletindo os valores possíveis para o limiar ϕ, passado como parâmetro para o *Tracking*. Esse valor de corte é aplicado sobre a contagem normalizada de uso de camadas, determinando um nível de relevância para a escolha dos regressores a serem utilizados na etapa de *Modelling*.

A escolha de valores baixos para ϕ implica que combinações camada/alvo que obtiveram menos votos durante as iterações sobre as diferentes partições de treino e validação serão utilizados na modelagem final. Esse fato tende a originar modelos com um maior número de regressores, uma vez que os modelos de camadas mais profundas, ou seja, mais distantes da primeira camada, tendem a trazer menores ganhos que os regressores das primeiras e, assim menos votos. Nesse caso, mesmo com menores quantidades de votos tais modelos serão escolhidos para compor os modelos de regressão finais para o DSTARS. Um raciocínio similar pode ser feito para valores de ϕ mais próximos de 1. Nesse caso, somente os modelos com as maiores quantidade de votos são escolhidos, implicando na escolha dos regressores que geraram ganhos na maioria das combinações treino/validação avaliadas. Frequentemente tais preditores se situam nas primeiras camadas de regressores, dando origem então a modelos DSTARS mais rasos.

Um exemplo de aplicação do limiar ϕ é apresentado na Tabela 2. Nessa situação hipotética, o problema tratado conta com três variáveis de saída, representadas por y_1, y_2 e y_3. Além disso, uma estratégia de particionamento por validação cruzada com 10 dobras foi utilizada sobre os dados passados para o DSTARS. No exemplo apresentado, a primeira porção da tabela (à esquerda) representa a contagem de votos que a combinação camada/*target* recebeu após a realização do Tracking. A porção central apresenta tais valores de voto após a normalização pelo número de partições empregado (10 dobras nesse caso). Por fim, a porção mais a direita apresenta quais camadas foram escolhidas para cada alvo a partir da aplicação de um limiar $\phi = 0.5$. As combinações marcadas como V (Verdadeiro) serão utilizadas na etapa de *Modelling*, enquanto as marcadas como F (Falso) serão ignoradas nesse passo.

Tabela 2 – Exemplo de aplicação do limiar ϕ em um exemplo hipotético

Camada	y_1	y_2	y_3		y_1	y_2	y_3		y_1	y_2	y_3
1	10	10	10		1.0	1.0	1.0		V	V	V
2	9	10	8	$\xrightarrow{\text{norm.}}$	0.9	1.0	0.8	$\xrightarrow{\phi=0.5}$	V	V	V
3	1	7	3		0.1	0.7	0.3		F	V	F
4	7	5	0		0.7	0.5	0.0		V	V	F

4.5 Modelling

A etapa final do DSTARS consiste na indução de regressores utilizando os conjuntos originais X e Y, empregando as camadas de regressores que foram determinadas no *Tracking* e as correlações entre as saídas medidas na etapa de *Filtering*. O Algoritmo 6 apresenta a etapa de *Modelling* do DSTARS. O procedimento apresentado recebe como parâmetros os conjuntos X e Y originais, que haviam sido previamente particionados, a descrição de utilização de camadas T, obtida na etapa de *Tracking*, e as informações sobre as dependências estatísticas entre os alvos, F, que foram obtidas na etapa de *Filtering*. A etapa final do DSTARS, de fato simples, segue as informações definidas em T: caso uma camada tenha sido escolhida para um determinado alvo t (linha 7), um regressor é induzido para Y_t com os atributos de entrada X aumentados com as predições das variáveis de saída, obtidas pelos regressores mais recentemente induzidos.

Dado um modelo DSTARS treinado, novas instâncias serão submetidas sequencialmente seguindo a ordem determinada para as camadas de regressores. Cada preditor na sequência gerará predições que agirão como atributos extras para os posteriores, sendo que os últimos regressores pertencentes às camadas de cada atributo alvo proverão as predições finais para as instâncias processadas.

Algoritmo 6 Algoritmo para etapa de *Modelling* do DSTARS

1: **Função** MODELLING(X, Y, T, F)
2: $dstars \leftarrow \{\}$
3: $\overset{aprox}{Y} \leftarrow \{\}$
4: **para** l $\leftarrow 1$ **até** nLinhas(T) **faça**
5: $\overset{aux}{Y} \leftarrow \overset{aprox}{Y}$
6: **para** t $\leftarrow 1$ **até** d **faça**
7: **se** $T[l,t] =$ **Verdadeiro então**
8: // Filtragem dos *targets* relevantes para t
9: $f \leftarrow F[t, 1..d]$
10: $\overset{tr}{X} \leftarrow X || \overset{aprox}{Y_f}$
11: // Indução de modelo ST
12: $mod : \overset{tr}{X} \rightarrow Y_t$
13: $\overset{aux}{Y_t} \leftarrow$ predição($mod, \overset{tr}{X}$)
14: $dstars_t^l \leftarrow mod$
15: **fim se**
16: **fim para**
17: $\overset{aprox}{Y} \leftarrow \overset{aux}{Y}$
18: **fim para**
19: **retorna** $dstars$
20: **fim Função**

4.6 Análise de Complexidade Computacional do DSTARS

A complexidade computacional total do DSTARS depende de certas escolhas realizadas durante a sua execução, como a estratégia de amostragem utilizada e os parâmetros ϕ e ε escolhidos. Quanto às etapas, o passo de *Filtering* possui complexidade $O(d\,r)$, onde d representa o número de alvos e r o custo do cálculo da métrica de importância escolhida, considerando cada um dos alvos. A etapa de *Tracking* possui complexidade $O(P\,L\,d\,b)$, onde P representa a quantidade de partições empregadas, L a quantidade máxima de camadas de regressores treinados até todos o *targets* atingirem a convergência, e b a complexidade do regressor base empregado. A complexidade do etapa de *Modelling* é $O(L\,d\,b)$.

Dessa forma, a complexidade total do DSTARS é dada por $O(d\,[r + (P+1)\,L\,b])$. Assim, o número de alvos do problema tratado, bem como a quantidade de partições empregadas influenciam grandemente na complexidade final do modelo. Os números de camadas obtidos durante a execução dos experimentos, apresentados na Seção 6.4, trazem maiores informações acerca do custo de treinamento do DSTARS em cenários reais.

5 MATERIAIS E MÉTODOS

O presente capítulo apresenta as bases de dados utilizadas, bem como as tecnologias, metodologias e métricas de avaliação utilizadas nos experimentos realizados para comparação do DSTARS com outras técnicas MT já existentes.

5.1 Base de Dados

Um total de dezoito bases de dados de regressão MT, previamente utilizadas em estudos anteriores [4, 3, 5, 8], foram utilizadas durante os experimentos como forma de comparar e avaliar o método DSTARS com as outras abordagens para tratar problemas MT consideradas nesse trabalho (ST, MTRS e ERC). As bases de dados utilizadas compreendem diferentes problemas de regressão com múltiplas saídas e áreas de aplicação. De fato, esse mesmo conjunto foi empregado no trabalho de Spyromitros-Xioufis et al. [5] onde os métodos MTRS e ERC foram propostos e, estão disponíveis juntamente com o software *Mulan*[1] de forma pública.

A Tabela 3 apresenta um sumário das informações sobre das bases de dados, descrevendo o número de instâncias, atributos e variáveis de saída de cada conjunto. Objetivando oferecer maiores detalhes acerca das bases de dados utilizadas, as subseções a seguir apresentam com maiores detalhes as características de cada conjunto, destacando o problema ao qual estão relacionadas. As informações relacionadas às bases de dados foram obtidas no trabalho de Spyromitros-Xioufis et al. [5].

5.1.1 andro

A base de dados *Andromeda* se refere à predição de seis atributos referentes à qualidade da água: temperatura, pH, condutividade, salinidade, oxigênio e nível de turbidez no golfo *Thermaikos* em *Thessaloniki*, Grécia. As medidas presentes na base foram tomadas por sensores com um intervalo de nove segundos entre cada amostragem, sendo que a média dos resultados foi calculada de forma a resultar em um único valor diário para cada variável.

5.1.2 atp1d e atp7d

As bases ATP (*Airline Ticket Price*) se referem à predição dos valores de passagens aéreas. As linhas correspondem a observações ordenadas por tempo em diversos dias. Cada

[1] <http://mulan.sourceforge.net/index.html>

Tabela 3 – Nome, número de exemplos, atributos e variáveis de saída das bases de dados utilizadas nos experimentos

Nome	**Exemplos**	**Atributos**	***Targets***
andro	49	30	6
atp1d	337	411	6
atp7d	296	411	6
edm	154	16	2
enb	768	8	2
jura	359	15	3
oes10	403	298	16
oes97	4334	263	16
osales	639	413	12
rf1	9125	64	8
rf2	9125	576	8
scm1d	9803	280	16
scm20d	8966	61	16
scfp	1137	23	3
sf1	323	10	3
sf1	1066	10	3
slump	103	7	3
wq	1060	16	14

amostra representa um conjunto de exemplos de uma data específica de consulta e uma data específica de saída. Os atributos nas bases incluem vários atributos numéricos e lógicos acerca das observações realizadas, os quais podem ser resumidos como: o número de dias entre a data de consulta e partida (1 atributo), variáveis lógicas correspondentes ao dia da semana da consulta (7 variáveis) e a enumeração completa dos quatro seguintes valores: (1) os valores mínimo e médio da passagem e o número de voos de (2) todas as companhias aéreas que aparecem em mais de 50% dos dias observados para (3) voos sem parada, com uma e duas paradas, levando em consideração (4) o dia da observação, um dia antes e dois dias antes. As variáveis alvo das bases de dados em questão se referem ao valor da passagem no próximo dia (*atp1d*) ou o preço mínimo levando em consideração os próximos sete dias (*atp7d*) para seis preferências de voo: (1) qualquer companhia com qualquer número de paradas, (2) qualquer companhia com voos sem parada e as companhias aéreas (3) *Delta*, (4) *Continental*, (5) *Airtrain* e (6) *United*.

5.1.3 edm

A base de dados *the Electrical Discharge Machining* representa um problema de regressão com duas saídas. A tarefa é buscar diminuir o tempo de usinagem simulando o comportamento de um operador de maquinário humano que controla o valor de duas variáveis. O problema em questão conta com variáveis contínuas para descrição e os valores de saída tomam três valores numéricos distintos ({-1,0,1}).

5.1.4 enb

A base de dados *Energy Building* se refere a predição dos requerimentos das cargas de aquecimento e resfriamento de construções (eficiência energética) em função de oito parâmetros de construção, tais como: área de vidraça, área do telhado, altura total, entre outros.

5.1.5 jura

A base de dados Jura consiste nas medições de concentração de sete metais pesados (Cádmio, Cobalto, Crômio, Cobre, Níquel, Chumbo e Zinco) no solo, tomadas em 359 localizações da região Suíça de Jura. O tipo de vegetação do local (Floresta, Pasto, Prado e Lavoura) e o tipo de rocha (*Argovian*, *Kimmeridgian*, *Sequanian*, *Portlandian* e *Quaternary*) também estão registrados, para cada local das medidas. De forma geral, o problema busca descrever a concentração de metais que são mais caros de serem medidos com base em metais cuja mensuração é mais barata. No caso, o Cádmio, Cobre e Chumbo são tratados como variáveis alvo, enquanto que o restante das informações são utilizados como atributos descritivos.

5.1.6 oes97 e oes10

As bases de dados relativas ao *Occupational Employment Survey* se referem às pesquisas anuais de ocupação profissional tomadas nos anos de 1997 (*oes97*) e 2010 (oes10) pelo serviço americano de estatísticas trabalhistas. Cada linha apresenta o número estimado de trabalhadores dentre vários tipos de empregos para uma região metropolitana específica. Nas pesquisas realizadas em 1997 e 2010, 334 e 403 cidades foram levadas em consideração, respectivamente. Os tipos de ocupação profissional apresentados nas bases de dados se referem a empregos que são observados em pelo menos 50% das cidades pesquisadas, de forma que certas categorias não possuem representantes para determinadas localidades. As variáveis alvo em ambas pesquisas são subconjuntos de profissões que seguem o mesmo requisito de constarem em pelo menos metade dos municípios presentes nos conjuntos avaliados. Vale ressaltar que valores ausentes nas bases (tanto de entrada, quanto de saída) foram substituídos pelo valor médio dos atributos.

5.1.7 osales

Esse conjunto de informações se refere a uma versão pré-processada da base de dados utilizada na competição *Kaggle's "Online Product Sales"*, que se refere à predição de vendas online de variados produtos. Cada amostra no conjunto representa um produto diferente, descrito por várias informações acerca de seu aspecto, além de características de sua campanha promocional. No problema em questão existem doze variáveis alvo, que correspondem às vendas mensais em um período de um ano após o lançamento de cada

produto. O pré-processamento consistiu na remoção de amostras com valores ausentes nas variáveis alvo, bem como a eliminação de atributos com apenas um valor distinto.

5.1.8 rf1 e rf2

As bases de dados *River Flow* se referem à predição do fluxo de redes fluviais em um período de 48 horas em localizações específicas. Os conjuntos contém informações coletadas com intervalos de uma hora de observações de fluxo em oito locais da rede fluvial do rio *Mississippi*, nos Estados Unidos, sendo obtidas pelo instituto *US National Weather Service*. Cada linha da base de dados contém a mais recente observação para cada um dos oito locais, bem como medições anteriores com intervalos de 6, 12, 18, 24, 36, 48 e 60 horas. Para cada local onde foram tomadas as medidas, 8 atributos descritivos foram coletados de forma a melhor explicar o problema. Para cada exemplo, as variáveis de saída correspondem ao nível de precipitação futuro em um período de 48 horas, com janelas de 6 horas (6, 12, 18, 24, 30, 36, 42 e 48 horas). Os conjuntos de dados são compostos por cerca de um ano de observações tomadas de hora em hora (mais de 9000 horas), coletadas no período entre setembro de 2011 e setembro de 2012. A base de dados rf2 estende a rf1 adicionando previsões dos valores esperados para cada ponto de medição e as mesmas janelas de tempo. Como ressalta Spyromitros-Xioufis et al. [5], este domínio é um candidato natural para regressão MT, tendo em vista as claras relações físicas entre as leituras no fluxo fluvial contínuo do rio.

5.1.9 scm1d e scm20d

As bases de dados *Supply Chain Management* (scm) são derivadas do torneio *Trading Agent Competition in Supply Chain Management* (TACSCM), realizado em 2010. O objetivo é a previsão de preços de produtos em intervalos futuros. Cada linha corresponde a um dia de observação no torneio, sendo que ao todo foram 220 dias para cada um dos 18 jogos realizados no campeonato. As variáveis de entrada correspondem aos preços observados em um dia específico da competição. Além disso, quatro observações atrasadas em relação ao tempo foram adicionadas para cada produto, de forma a facilitar a identificação de tendências de preços. Os intervalos levados em consideração foram 1, 2, 4 e 8 dias anteriores ao dia em questão. Cada um das dezesseis saídas do problema corresponde ao preço médio dos produtos simulados, levando em consideração o próximo dia (scm1d) ou os próximos vinte dias (scm20d).

5.1.10 scfp

Corresponde a uma versão pré-processada da base de dados utilizada na competição da *Kaggle* intitulada "*See Click Predict Fix*", que foi realizada em 2013. As três variáveis alvo do problema correspondem ao número de visualizações, cliques e comentá-

rios que chamadas especificas a serviços não-emergenciais receberam. Os chamados foram coletados de quatro cidades (Oakland, Richmond, New Haven, Chicago) dos Estados Unidos da América durante todo o ano de 2012. A versão utilizada corresponde à amostragem de 1% dos dados utilizados na competição. Os atributos descritivos são o número de dias que o chamado ficou *online*, o dispositivo no qual foi criado, o tipo de chamado (por exemplo, grafitagem, lixo, buracos em estradas, etc.), as coordenadas geográficas do chamado e sua distância para o centro da cidade, bem como a própria identificação do município do chamado. As variáveis categóricas foram transformadas em valores binários e os atributos resultantes que assumiram valores verdadeiro em menos de 1% dos casos foram removidos [5].

5.1.11 sf1 e sf2

As bases de dados *Solar Flare* apresentam três variáveis alvo correspondentes ao número de vezes que três tipos de explosões na superfície do sol foram observadas. Os tipos são Comum, Moderada e Severa, sendo observadas em um intervalo de 24 horas. Existem duas versões para essa descrição de problema: a base de dados sf1 contém mensurações realizadas em 1969, enquanto que a variante sf2 registra as explosões em um dia específico no ano de 1978.

5.1.12 slump

A base de dados *Concrete Slump* está relacionada à predição de três propriedades do concreto (consistência, escoamento e força compressiva). As variáveis alvo são definidas em função de sete ingredientes do concreto: cimento, cinzas suspensas, resíduos de queima, água, super-plastificante e agregados grosso e fino.

5.1.13 wq

A base de dados *Water Quality* possui quatorze atributos alvo que se referem à representação relativa de espécies animais e vegetais em rios da Eslovênia. Essas variáveis resposta estão relacionadas a dezesseis atributos descritivos que correspondem a atributos físicos e químicos de qualidade da água.

5.2 R e configurações dos algoritmos de regressão base

A suíte R oferece ferramentas para análise algébrica e estatística, além de vasta biblioteca para mineração de dados e aprendizado de máquina. Em adição, variadas formas de visualização de dados estão presentes na referida tecnologia, tornando o R uma plataforma de programação completa para desenvolvimento e análise de problemas rela-

cionados ao aprendizado de máquina. Em todos os experimentos realizados, a linguagem R foi empregada, inclusive no desenvolvimento do DSTARS.

A Tabela 4 apresenta os algoritmos de regressão base empregados nos experimentos e os respectivos pacotes R utilizados. Vale ressaltar que em todos os casos foram utilizadas as configurações padrão dos algoritmos de regressão, uma vez que se buscou realizar uma comparação igualitária entre todas as técnicas MT. Assim, foge do escopo desse trabalho avaliar o impacto de diferentes configurações dos algoritmos base, ainda que uma escolha adequada de hiper-parâmetros possa melhorar o desempenho obtido significantemente. Essa análise constitui um caminho futuro para pesquisas.

Tabela 4 – Algoritmos de regressão utilizados nos experimentos e os respectivos pacotes R

Algoritmo	**Pacote R**
Support Vector Machine (SVM)	e1071
Random Forest (RF)	ranger
Extreme Gradient Boosting (XGBoost)	xgboost
Classification and Regression Tree (CART)	rpart

5.3 Descrição dos experimentos realizados e parametrização do DSTARS empregada

Durante os experimentos realizados, um processo de validação cruzada com 10 dobras foi empregado para particionar todas as bases de dados em conjuntos de treino e teste, como frequentemente reportado na literatura [2, 23, 5, 8]. Similarmente, na etapa *Tracking* do DSTARS, a estratégia de validação cruzada, também com 10 dobras, foi empregada para determinar o número mais adequado de camadas de regressores para cada alvo.

Visando verificar qual seria o impacto da escolha do hiper-parâmetro ϕ no desempenho final do DSTARS, um procedimento de *tuning* de parâmetro foi realizado. Tendo em vista a escolha da estratégia de validação cruzada com 10 dobras para o *Tracking*, cada combinação camada/*target* poderia acarretar na redução do erro preditivo em no máximo o número de partições consideradas, ou seja, dez casos (nos experimentos realizados). Por essa razão, todos os valores possíveis de corte na escolha de camadas foram considerados. Por um critério de padronização, a contagem de casos em que uma camada gerou diminuições no erro para cada alvo foi normalizada no intervalo de 0 a 1. Para tal, os valores de contagem obtidos foram divididos pelo número de dobras consideradas na validação cruzada.

Como já mencionado no Capítulo 4, o limiar ϕ determina o grau mínimo de contribuição que uma camada deve trazer para um alvo específico. Dessa forma, valores baixos

de ϕ implicam que camadas com menos votos poderão ser utilizadas no treinamento dos modelos finais do DSTARS, o que tende a acarretar na utilização de mais camadas de regressores por alvo. Em contrapartida, valores mais altos de ϕ implicam na escolha de camadas que obtiveram mais votos, o que está intrinsecamente relacionado ao uso de menos camadas. Nos experimentos foram avaliados valores pertencentes ao intervalo de $[0, 1]$ com uma granularidade de 0.1.

Em adição, o parâmetro ε, relativo ao critério de parada na adição de novas camadas de regressores durante o *Tracking*, foi mantido em 10^{-4} durante todos os experimentos. Assim, considerou-se como irrelevante o ganho resultante da adição de um novo regressor caso a redução do erro preditivo trazida por este afetasse a métrica de erro empregada a partir apenas do quarto dígito decimal. Dessa forma, assume-se que as poucas melhorias obtidas não justificam o custo computacional adicional decorrente da adição de um novo regressor. No entanto, a exploração de outros valores para ε é um campo para futura exploração.

Ademais, os quatro métodos para resolução de problemas MT avaliados (ST, MTRS, ERC e DSTARS) foram executados utilizando as dezoito bases de dados de *benchmark* anteriormente apresentadas. Para cada combinação de método MT e base de dados os quatro algoritmos de regressão previamente mencionados (RF, SVM, XGboost e CART) foram executados, visando analisar como a escolha de diferentes preditores base impactaria no desempenho obtido pelos modelos para MTR.

5.4 Métricas de Avaliação

Para a avaliação dos métodos MT, três diferentes métricas de desempenho foram empregadas: o erro quadrático médio relativo (RRMSE), a média dos erros quadráticos médios relativos (aRRMSE) e a performance relativa (PR).

O valor do erro quadrático médio relativo ou *Relative Root Mean Squared Error* (RRMSE) é obtido a partir do cálculo da diferença quadrática entre os valores reais de um *target* e os valores preditos por um regressor, divididos pela diferença entre os valores reais do mesmo alvo e sua média. Esse último termo médio age como uma *linha base* da métrica, que permite a mensuração do ganho obtido pelo modelo treinado sobre um preditor simples, que sempre prediz um valor constante (média da variável alvo). Essa métrica é muito útil para comparar variáveis de saída que seguem distribuições não-homogêneas, e tem sido empregada como medida de avaliação em diversos trabalhos em MTR [4, 5, 8]. A Equação 5.1 apresenta o cálculo do RRMSE considerando N instâncias.

$$\text{RRMSE} = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y}_i)^2}} \tag{5.1}$$

Na equação, y representa os reais valores da i-ésima instância de uma variável de saída, $\hat{y}$ representa os valores preditos pelo modelo treinado e, por fim, $\overline{y}$ é média dos N valores de y.

O RRMSE é calculado para cada uma das d variáveis alvo em problema MT. Para se obter uma métrica unificada de desempenho para todas as respostas e assim, um indicativo geral de erro preditivo, a média dos d valores de RRMSE é calculada, resultando na métrica aRRMSE [4, 5, 8]. O cálculo do *average Relative Root Mean Squared Error* (aRRMSE) é apresentado na Equação 5.2.

$$\text{aRRMSE} = \frac{1}{d}\sum_{t=1}^{d}\sqrt{\frac{\sum_{i=1}^{N}(y_i^t - \hat{y}_i^t)^2}{\sum_{i=1}^{N}(y_i^t - \overline{y}_i^t)^2}} \tag{5.2}$$

Por fim, a *Performance* Relativa (PR) compara o valor de aRRMSE de um modelo ST com o aRRMSE obtido por outro método MT M (no caso, com o MTRS, ERC ou DSTARS), levando em conta uma base de dados. Dessa forma, essa métrica indica se houve melhoria (PR maior que 1) ou degradação (PR menor que 1) em relação ao desempenho obtido pela estratégia ST [5]. A Equação 5.3 apresenta a fórmula de cálculo da PR.

$$PR_d(M) = \frac{\text{aRRMSE}(ST)}{\text{aRRMSE}(M)} \tag{5.3}$$

Em adição, as métricas definidas permitem a investigação da possível superioridade de uma técnica MT sobre as outras através da aplicação de um teste estatístico. Nesse trabalho o teste estatístico de *Friedman* foi adotado, utilizando um nível de significância de $\alpha = 0.05$. A hipótese nula consiste na afirmação de que o desempenho de todas as técnicas MT são estatisticamente equivalentes, levando em consideração os critérios de comparação adotados. Quando o *p-valor* obtido no teste Friedman é menor que o nível de significância adotado, a hipótese nula é rejeitada, implicando que existem diferenças estatísticas entre os métodos comparados.

Quando diferenças estatísticas são observadas, o teste *post hoc* de *Nemenyi* pode ser aplicado, discriminando que o desempenho de dois modelos é significantemente diferente se seus correspondentes *ranks* de desempenho médios diferem por pelo menos um valor de Distância Crítica (*Critical Distance* - CD). Quando vários modelos são comparados dessa forma, uma representação gráfica pode ser utilizada para representar os resultados através de um Diagrama de Distância Crítica, como proposto no trabalho de Demšar [38].

6 RESULTADOS E DISCUSSÃO

Esse capítulo descreve e discute os resultados obtidos quando comparando o desempenho das quatro abordagens para problemas MT consideradas (ST, MTRS, ERC e DSTARS) combinadas com os regressores e bases de dados utilizados. Primeiramente, os resultados obtidos são discutidos com relação ao erro preditivo. Em seguida, os três métodos para regressão MT (MTRS, ERC e DSTARS) são comparados tendo em vista o seu ganho de desempenho relativo ao ST. Os resultados de testes estatísticos em diferentes cenários são também apresentados, de forma a verificar a possibilidade de diferença significante dentre as abordagens MT. Por fim, uma análise detalhada do desempenho do DSTARS nos problemas avaliados é realizada, levando em consideração o número de camadas de regressores que foram utilizadas para cada base de dados, os melhores valores de ϕ encontrados durante o *tuning* de parâmetro utilizado e as diferenças de desempenho observadas variando-se o parâmetro que foi ajustado.

6.1 Análise dos erros obtidos

Na Tabela 5 estão reportados os valores médios de aRRMSE que foram obtidos a partir da execução de todas as combinações entre abordagens MT, regressores e bases de dados, utilizando os procedimentos descritos na Seção 5.3. Nesta, para cada combinação regressor/base de dados, o método MT que originou o menor erro preditivo está em negrito. Similarmente, o menor erro obtido para cada base de dados, independentemente do regressor empregado, está sublinhada. Levando em conta as bases de dados e regressores utilizados, 72 resultados foram reportados por cada método MT. Assim, setenta e dois representa o número máximo de vezes que um método MT poderia obter os melhores resultados nessa análise.

Tabela 5 – Resultados de aRRMSE considerando todos os métodos MT, regressores e base de dados avaliados. Os melhores resultados para cada combinação regressor/base de dados estão em negrito. Os melhores resultados por base de dados estão sublinhados

Dataset	Regressor	ST	MTRS	ERC	DSTARS
atp1d	RF	0.3920	0.3904	0.3902	**0.3889**
	SVM	**0.4397**	0.4402	0.4398	**0.4397**
	XGBoost	0.4048	0.4080	**0.3830**	0.4042
	CART	0.4717	0.4843	**0.4683**	0.4704

Dataset	Regressor	ST	MTRS	ERC	DSTARS
atp7d	RF	**0.5164**	0.5169	0.5178	0.5167
	SVM	**0.6404**	0.6414	0.6410	**0.6404**
	XGBoost	0.5068	0.5228	**0.4751**	0.5020
	CART	0.6979	0.6805	**0.6570**	0.6889
oes97	RF	0.5164	0.5135	**0.5133**	0.5138
	SVM	0.6118	0.6123	**0.6110**	0.6111
	XGBoost	0.5779	0.5886	**0.5673**	0.5773
	CART	0.6971	0.6956	0.6984	**0.6910**
oes10	RF	0.4070	0.4081	0.4070	**0.4057**
	SVM	0.5464	0.5456	**0.5451**	0.5456
	XGBoost	0.4691	0.4754	**0.4611**	0.4691
	CART	0.5976	0.6026	**0.5971**	0.5976
rf1	RF	0.0782	0.0582	0.0731	**0.0534**
	SVM	0.1215	0.1070	0.1151	**0.0997**
	XGBoost	0.1274	0.1340	**0.1150**	0.1245
	CART	0.3523	0.3510	**0.3505**	0.3510
rf2	RF	0.0847	0.0784	0.0852	**0.0753**
	SVM	0.1095	0.1064	0.1084	**0.1063**
	XGBoost	0.1293	0.1320	**0.1212**	0.1282
	CART	0.3594	0.3720	**0.3552**	0.3568
scm1d	RF	0.2871	0.2758	0.2823	**0.2716**
	SVM	0.3309	0.3232	0.3258	**0.3210**
	XGBoost	0.3059	0.3074	**0.2802**	0.3056
	CART	0.5104	0.5029	**0.4867**	0.4985
scm20d	RF	0.3648	0.3313	0.3347	**0.3116**
	SVM	0.3972	0.3522	0.3474	**0.3301**
	XGBoost	0.3701	0.3625	**0.3262**	0.3625
	CART	0.7307	0.7107	**0.6772**	0.7010
edm	RF	0.6721	0.6631	0.6661	**0.6478**
	SVM	0.7699	0.7714	0.7667	**0.7646**
	XGBoost	0.6793	0.7224	**0.6581**	0.6793
	CART	0.7042	0.7379	**0.6850**	0.7042
sf1	RF	1.0051	1.1280	1.0161	**1.0049**
	SVM	0.9390	0.9477	**0.9250**	0.9390
	XGBoost	1.3598	1.3479	**1.2478**	1.3374
	CART	**0.9667**	1.0278	0.9723	**0.9667**

Dataset	Regressor	ST	MTRS	ERC	DSTARS
sf2	RF	0.8487	0.9410	0.8617	**0.8479**
	SVM	0.7825	0.7787	0.7827	**0.7764**
	XGBoost	1.0609	1.1126	**1.0109**	1.0609
	CART	0.8305	**0.8243**	0.8247	0.8302
jura	RF	0.6061	0.5974	0.5969	**0.5921**
	SVM	**0.6409**	0.6413	0.6391	**0.6409**
	XGBoost	0.6383	0.6425	**0.6129**	0.6383
	CART	**0.7021**	0.7240	0.7042	**0.7021**
wq	RF	0.9066	0.9392	**0.9059**	0.9068
	SVM	0.9630	**0.9535**	0.9581	0.9549
	XGBoost	0.9828	1.0263	**0.9429**	0.9828
	CART	0.9786	1.0040	**0.9472**	0.9781
enb	RF	0.1504	0.1173	0.1304	**0.1161**
	SVM	0.2499	0.2173	0.2404	**0.1678**
	XGBoost	0.0601	0.0617	0.0744	**0.0599**
	CART	**0.2906**	0.2971	0.2909	**0.2906**
slump	RF	0.8365	0.8324	**0.8222**	0.8223
	SVM	0.6924	0.6862	0.6819	**0.6784**
	XGBoost	0.8518	0.8575	**0.7975**	0.8508
	CART	1.0254	1.0376	**1.0016**	1.0252
andro	RF	0.7941	0.7349	0.7614	**0.6713**
	SVM	1.1348	0.9243	1.0089	**0.7561**
	XGBoost	0.4754	0.4705	**0.4130**	0.4676
	CART	0.6677	0.6343	**0.6118**	0.6284
osales	RF	0.7577	**0.7275**	0.7332	0.7284
	SVM	1.1726	**1.1685**	1.1702	**1.1685**
	XGBoost	0.8404	0.8695	**0.7679**	0.8404
	CART	0.9560	0.9694	**0.7920**	0.9407
scfp	RF	0.9263	0.9379	**0.8778**	0.8939
	SVM	0.8242	0.8256	**0.8151**	0.8242
	XGBoost	1.1163	1.1422	**0.9248**	1.1067
	CART	1.1075	1.0658	1.0882	**1.0214**

Como pode ser observado a partir da Tabela 5, nos experimentos realizados, frequentemente os regressores *ensemble* obtiveram os menores resultados de erro. O uso da SVM também apresentou os melhores desempenhos em alguns casos, todavia, o uso de um modelo mais simples como a CART não resultou na obtenção dos menores preditivos,

como esperado. Em relação às abordagens MT, considerando todos os regressores, o método ST obteve os menores de erro em 7 casos. O MTRS por sua vez, obteve os menores aRRMSEs em 4 quatro casos. O ERC e o DSTARS, com maior relevância, originaram os menores valores de aRRMSE em 37 e 31 casos, respectivamente. A soma das contagens superar setenta e dois decorre da ocorrência de empates entre as abordagens MT em alguns casos.

Apesar de aparente superioridade do ERC frente ao DSTARS, uma análise mais profunda deve ser realizada. A Tabela 6 apresenta a contagem de casos em que cada método MT obteve o menor aRRMSE para cada regressor. Como é possível perceber, o DSTARS obteve a maioria dos menores erros quando combinado com a RF e a SVM. No entanto, para o XGBoost e CART o cenário foi dominado pelo ERC, o que justifica os 37 casos de menores erros apresentados anteriormente. As abordagens ST e MTRS obtiveram pouco destaque nessa análise.

Tabela 6 – Número de vezes que cada método MT obteve o menor valor de aRRMSE por regressor. Para cada variação o melhor método MT está destacado

MT/Regressor	RF	SVM	XGBoost	CART	Todos
ST	1	3	0	3	0
MTRS	1	2	0	1	1
ERC	4	4	**17**	**12**	7
DSTARS	**12**	**13**	1	5	**10**

Esse tipo de resultado mostra que os métodos MT se comportam de forma diferente dependendo do regressor empregado. A superioridade observada na combinação do ERC com o XGBoost pode ser compreendida a partir da análise da natureza do regressor empregado. O XGBoost tem como a base a combinação de múltiplos modelos preditivos, no caso árvores de regressão, que são adicionados sequencialmente através do procedimento de *boosting*. Neste, cada novo preditor inserido no *ensemble* tenta diminuir o erro preditivo dos anteriores. Apesar de muito flexível e customizável, o uso de *boosting* tende a apresentar gerar sobreajuste aos dados [39]. Em outras palavras, durante o treinamento, os modelos adicionados minimizam o erro de predição no conjunto de treinamento empregado, sendo que este comumente não reflete todas as características do problema tratado, levando a um desempenho insatisfatório do modelo preditivo em outros cenários que não sejam para os quais este foi treinado.

O XGBoost conta com um grande número de parâmetros de configuração que visam minimizar a ocorrência de sobreajuste aos dados de treinamento. Tais parâmetros são relacionados tanto a customizações dos componentes individuais do conjunto (árvores), quanto a ajustes globais do *ensemble* [6]. No entanto, como já mencionado anteriormente, nos experimentos realizados, os valores padrão de configuração dos regressores foram mantidos, uma vez que o enfoque era nos métodos MT. Acontece que as características do

ERC podem ter lhe dado uma vantagem competitiva em relação aos outros métodos avaliados: diferente das outras abordagens (ST, MTRS e DSTARS), o ERC por si só também é um *ensemble*, combinando as respostas de suas múltiplas cadeias aleatórias de variáveis alvo através da média (vide Seção 3.2.2). O resultado desse fato é que a combinação das predições de múltiplos modelos (possivelmente) sobreajustados a diferentes conjuntos de treinamento (conjuntos aumentados com as predições dos targets), tende a suavizar a disparidade ou variabilidade existente entre as respostas, levando a menores erros preditivos. As outras abordagens MT não contam com esse mecanismo de agregação interno e, por essa razão, poderiam ter sido prejudicadas quando aliadas com o XGBoost. Uma etapa prévia de configuração desse regressor poderia minimizar os problemas apresentados, mas como já mencionado, foge do escopo desse trabalho a avaliação das particularidades dos regressores empregados.

Ainda que o ERC tenha sido hegemônico em relação ao ranqueamento dos erros quando combinado ao XGBoost, esse fato não é um sinônimo de obtenção dos menores erros preditivos na maioria dos casos. Ainda referindo-se a Tabela 6, sua última coluna apresenta a contabilização das vezes que cada abordagem MT obteve os menores erros preditivos em cada uma das dezoito bases de dados avaliadas, independentemente do regressor empregado (resultados correspondentes aos elementos sublinhados da Tabela 5). Como pode ser observado, o DSTARS obteve os menores índices de erros em 10 casos, contra 7 do ERC e 1 do MTRS. De fato, analisando a Tabela 5 é possível constatar que os menores aRRMSE foram atingidos em 10 casos pela RF, contra 4 casos da SVM e 4 casos do XGBoost.

Assim, é correto afirmar que a SVM e principalmente a RF resultaram nos modelos preditivos mais acurados de forma geral, quando combinados ao DSTARS. Para uma análise detalhada do desempenho preditivo de cada método MT e regressor, levando em conta cada *target* das dezoito bases de *benchmark* empregadas de forma individual, o leitor pode se referir à Tabela 10, presente no Apêndice A (*Resultados detalhados dos Experimentos realizados*).

Em relação aos resultados obtidos, nos casos onde os menores erros preditivos foram alcançados pelo DSTARS, espera-se que as saídas dos problemas tratados apresentem maiores níveis de correlação entre si. De forma similar, quando o desempenho do DSTARS se aproximar da estratégia ST, espera-se que as variáveis de saída do problema tratado não estejam correlacionadas entre si ou possuam baixos níveis de dependência estatística. A seguir a análise de correlação entre os alvos de dois dos problemas tratados são apresentadas, utilizando os valores médios de importância de variáveis que foram medidos pelo DSTARS, utilizando os procedimentos descritos na Seção 4.3. O primeiro caso apresentado corresponde à base de dados *sf1*, que contém três variáveis de saída, como apresentado na Seção 5.1. Nesse caso, os valores de aRRMSE obtidos para todos métodos

MT e regressores avaliados foram altos. De fato, a estratégia ST foi superior aos métodos MT em grande parte dos casos para essa base de dados. O segundo caso apresentado corresponde ao conjunto *rf1*, que possui oito variáveis de saída. Para a *rf1*, os métodos MT originaram menores valores de aRRMSE quando comparados à estratégia ST, independentemente do regressor empregado. O DSTARS obteve os menores valores de aRRMSE nesse conjunto, quando combinado à RF. Tendo em vista que a RF foi apontada como o regressor com os melhores resultados no geral, nas análises a seguir, os resultados obtidos por essa técnica são reportados.

Os valores médios (dentre todas as partições da validação cruzada avaliadas) de `RFimp` obtidos pelo DSTARS para a base de dados *sf1* são apresentados na Figura 5. Na figura, todas as possíveis combinações dos alvos para o cálculo da `RFimp`, considerando o problema em questão, são apresentadas. O valores de `RFimp` apresentados em cada uma das células da matriz representam os níveis de importância dos alvos indicados nas colunas, ao serem empregados como fonte de informação para descrever as variáveis assinaladas nas linhas. Vale relembrar que os valores da métrica apresentada representam a porcentagem de aumento no OOBE obtido pela RF para cada um dos *targets*, quando estes tiveram seus valores aleatoriamente permutados e foram usados para predizer os valores dos outros.

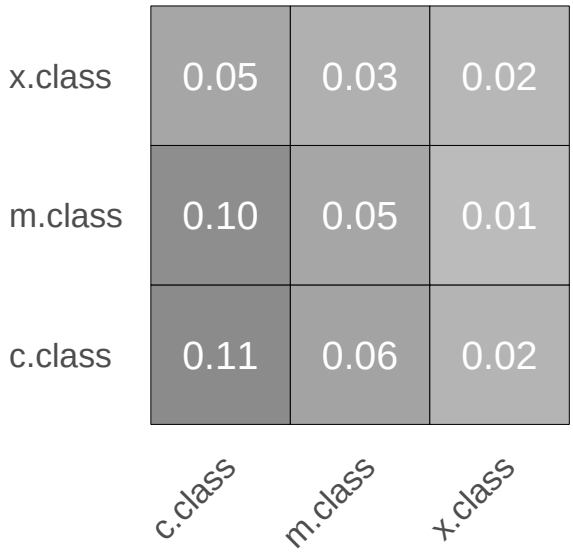

Figura 5 – Valores de importância de variáveis da RF da base de dados *sf1*, mensurados durante a execução do DSTARS. A estratégia ST apresentou superioridade em desempenho preditivo frente aos métodos MT nesse conjunto de dados

É possível perceber que os valores de importância observados foram todos baixos, no melhor caso correspondendo a 0.11. Assim, para o caso específico das aproximações do alvo *c.class* sendo utilizadas como fonte de informação para explicar essa mesma resposta, a permutação das predições ST do *target* em questão fez com que o OOBE aumentasse em 11%. A perturbação do OOBE ocasionada pela permutação dos valores preditos por

regressores ST para os outros alvos foi ainda menor. Assim, a ausência de ganhos pelo DSTARS, e pelos outros métodos MT de forma geral, é justificada pelo baixo nível de correlação entre as variáveis de saída do problema tratado.

Os valores de `RFimp` para o segundo caso apresentado, correspondente à base de dados *rf1*, são apresentados na Figura 6. Nesse conjunto de dados, os valores de `RFimp` observados são maiores, um indicativo de que as correlações existentes entre os alvos são mais proeminentes. No entanto, os níveis de dependência entre as respostas foram diferentes, havendo casos onde a permutação de uma das respostas degradou o OOBE em quase 30% (alvo *CLKM7* explicando o *VALI2*) e, em outros, o impacto foi de 5% (alvo *NASI2* explicando o *CLKM7*). De fato, o *target NASI2* foi pouco relevante para todos os outros alvos na análise apresentada. Os diferentes níveis de dependência estatística observados para o problema em questão justificam a razão pela qual o DSTARS foi capaz de obter os menores valores de aRRMSE, uma vez que o método proposto foi desenvolvido para lidar com esse tipo de situação.

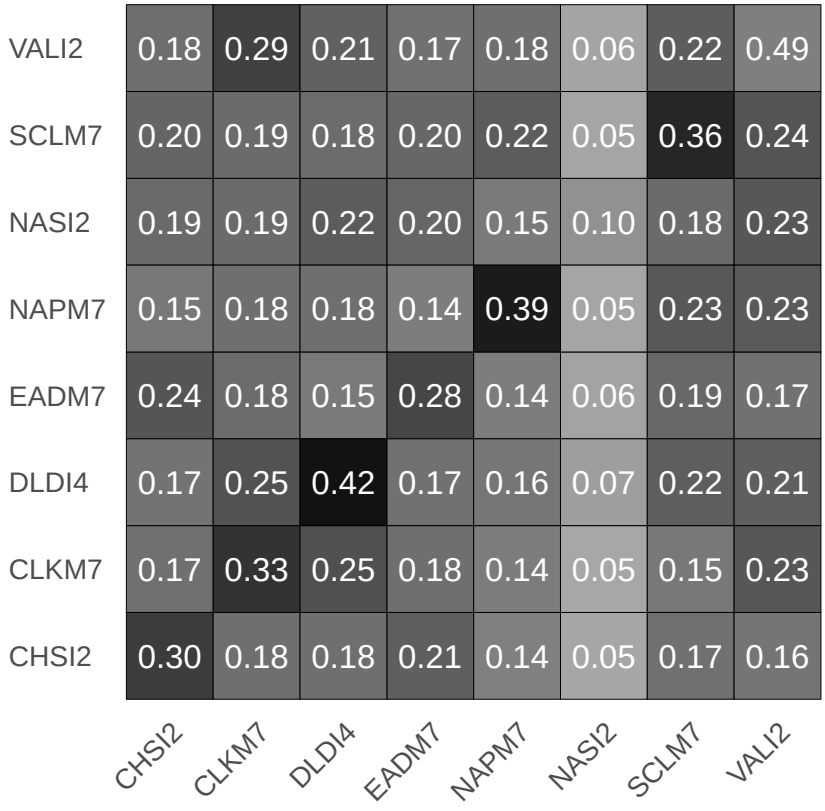

Figura 6 – Valores de importância de variáveis da RF da base de dados *rf1*, mensurados durante a execução do DSTARS. O DSTARS obteve os menores valores de erro nesse conjunto, quando combinado à RF

6.2 Comparação dos métodos multi-target com a abordagem single-target

De forma a avaliar os ganhos trazidos pelos métodos de regressão MT sobre a abordagem ST, a métrica PR (Performance Relativa) foi empregada. A Figura 7 apresenta os resultados obtidos considerando essa métrica. Na figura, cada gráfico de radar representa um dos regressores utilizados. Em adição, cada vértice representa uma das bases de dados avaliadas e cada polígono um método MT. Tendo em vista que os resultados reportados se relacionam à PR, valores menores do que 1 indicam que uma abordagem MT foi inferior ao ST. Similarmente, um PR maior do que 1 mostra a superioridade de um método MT frente ao ST.

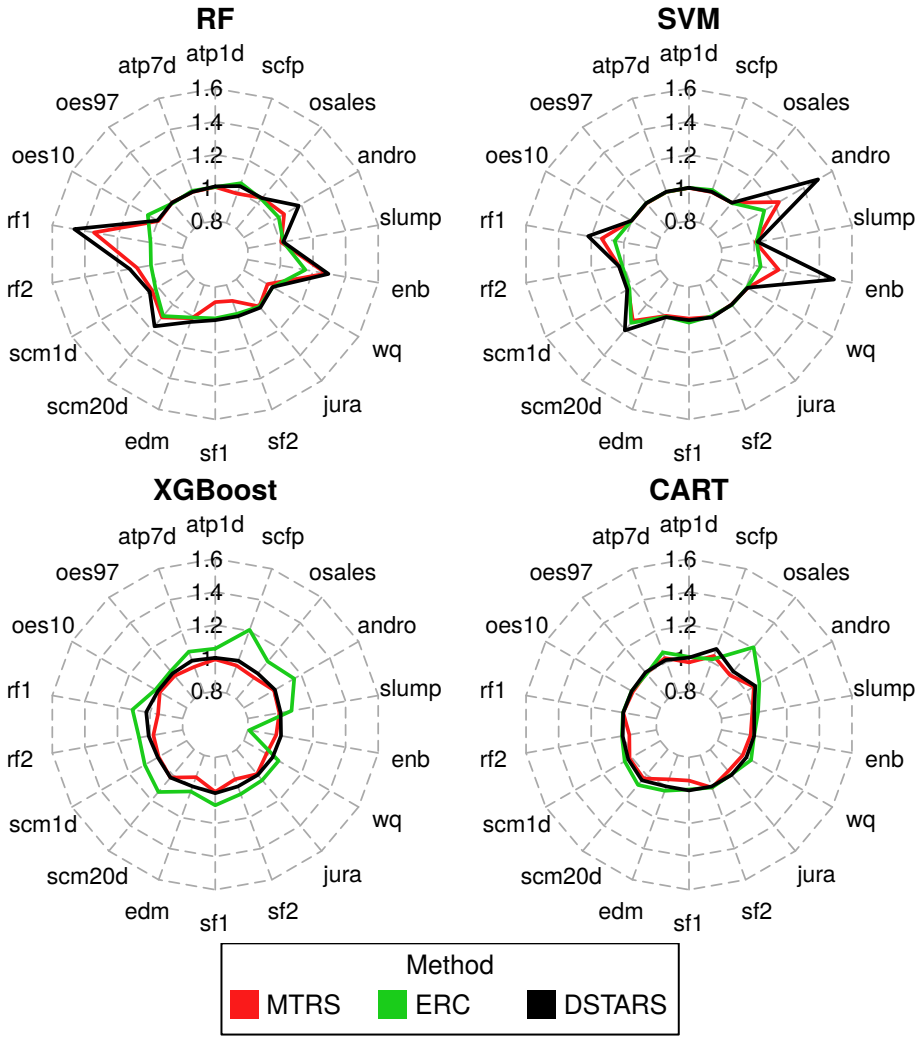

Figura 7 – Ganhos obtidos pelo uso das abordagens MT mensurados através da PR por base de dados. Quanto maior o polígono do método MT, maior o ganho obtido pela abordagem

Como já constatado na seção anterior, considerando os regressores RF e SVM, o polígono correspondente ao DSTARS apresentou maior área, dominando em praticamente todos os cenários as outras abordagens. Além disso, é possível perceber casos onde o

ganho do DSTARS em relação ao ST (e mesmo em relação aos outros métodos MT) foi claramente mais pronunciado. As bases de dados *rf1*, *enb* e *andro* podem ser citadas como exemplos desses ganhos. O mesmo cenário não foi observado para o XGBoost e a CART. Nos gráficos de radar correspondentes a esses regressores, salvo alguns casos onde o ERC trouxe ganhos mais pronunciados, praticamente todos os métodos MT tiveram desempenhos muito próximos à abordagem ST.

6.3 Testes estatísticos realizados

Testes estatísticos de Friedman, seguidos pelo teste *post hoc* de Nemenyi foram aplicados em diferentes cenários para avaliar o desempenho dos métodos MT. Primeiramente, os valores de aRRMSE foram considerados para verificar se alguma combinação regressor/método MT teria desempenho significantemente superior aos outros. Os resultados dessa avaliação estão apresentados na Figura 8. Variantes conectadas pela barra horizontal, representando a distância crítica obtida no teste, não são significantemente diferentes.

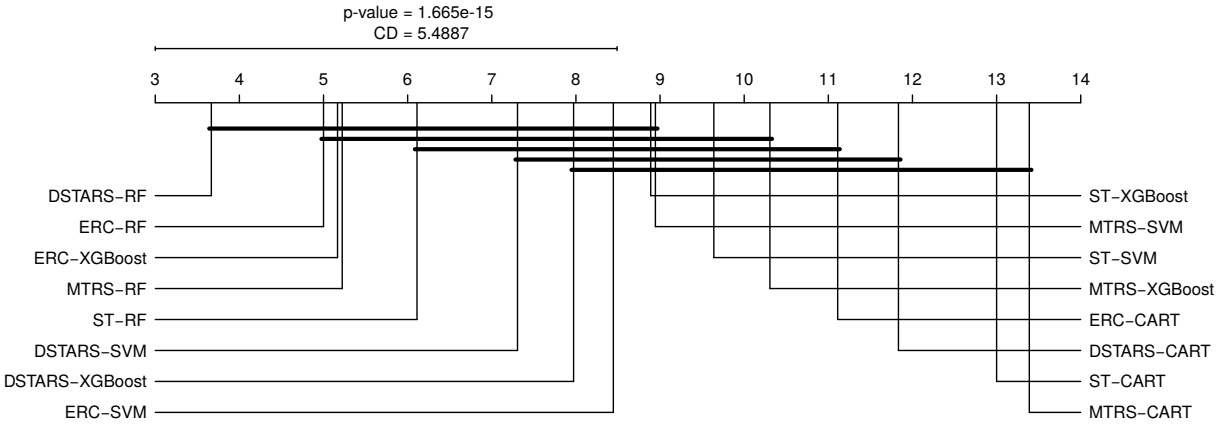

Figura 8 – Resultados do teste de Friedman e do teste *post hoc* de Nemenyi com $\alpha = 0.05$ para os valores de aRRMSE por base de dados, considerando todos métodos MT e regressores

Como pode ser observado, o primeiro grupo obtido no teste contém primeiramente a combinação DSTARS-RF, seguida, após um intervalo de quase um ponto no *ranking* médio de desempenho, pela combinação ERC-RF. A combinação ERC-XGBoost aparece em seguida, atestando os resultados apresentados anteriormente, de que o fato do ERC gerar os menores erros com o XGBoost não significa que esta é a combinação com menor aRRMSE dentre todas as avaliadas. Considerando a SVM, o DSTARS novamente é o método MT com menores resultados de erros. Como esperado, o último grupo conectado por um valor de CD contém em suas últimas posições métodos combinados com o regressor CART. É possível notar também que a abordagem ST foi superada em quase todos os casos pelas abordagens MT, quando utilizando os mesmos regressores.

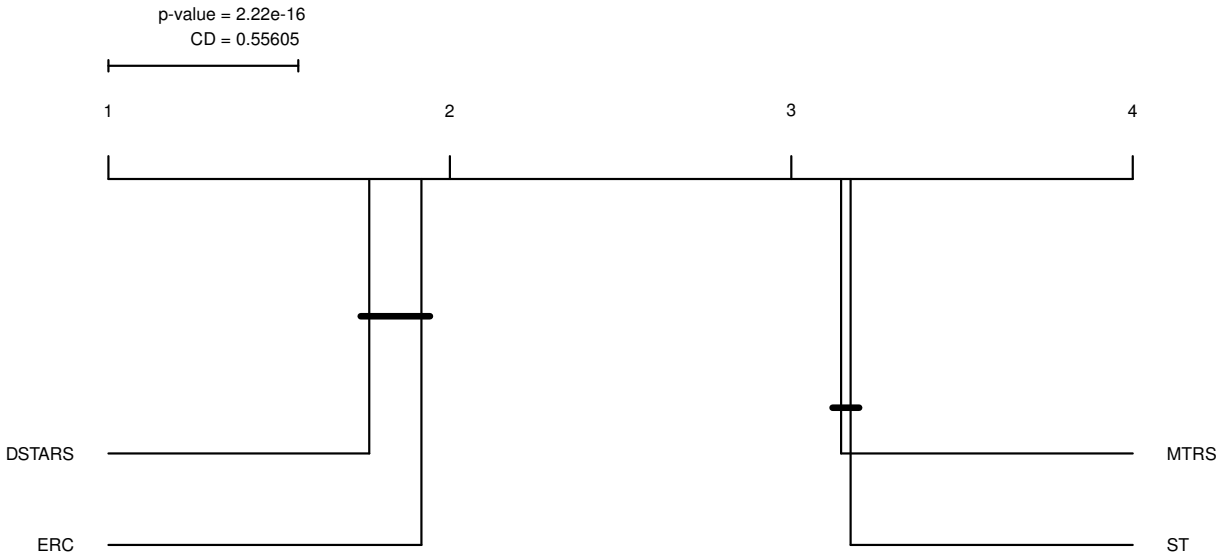

Figura 9 – Resultados do teste de Friedman e do teste *post hoc* de Nemenyi com $\alpha = 0.05$ para os valores de aRRMSE por base de dados, considerando todos métodos MT, independentemente do regressor utilizado

A seguir, uma análise considerando os erros obtidos pelos métodos MT independentemente do regressor empregado foi realizada. Nesse teste, os resultados de erro obtidos por todos os regressores foram contabilizados para cada método MT. Novamente os valores de aRRMSE foram considerados. Os resultados obtidos nessa variante são apresentados na Figura 9. Dessa vez se torna mais clara a diferença entre as abordagens MT avaliadas. No primeiro grupo se situam o DSTARS e o ERC, não diferindo estatisticamente. Em seguida vêm o MTRS e o ST, situados muito próximos no ranqueamento. A proximidade constatada entre o DSTARS e o ERC se deve ao fato do teste estatístico levar em consideração o *ranking* dentre as variantes comparadas em cada uma das entradas avaliadas, sendo que tais entradas corresponderam ao desempenho de um regressor em uma base de dados para cada uma das abordagens MT. Dessa forma, a proeminência de melhorias do ERC quando aliado à CART e ao XGBoost auxiliaram esse método MT a aumentar sua posição na ordenação dos melhores métodos.

A próxima análise realizada considerou o RRMSE obtido por *target*, levando em conta todos as combinações entre métodos MT e regressores. Assim, cada entrada para o teste apresentava o erro para um alvo específico em uma base de dados. A Figura 10 apresenta os resultados computados para essa variante de testes. Na figura é clara a separação estatisticamente significante (com $\alpha = 0.05$) entre o primeiro grupo, composto sequencialmente por DSTARS-RF, MTRS-RF, ERC-RF, ERC-XGBoost e ST-RF, e as demais possibilidades. É válido notar que o DSTARS aparece também nas primeiras posições do segundo grupo conectado.

Em seguida, uma análise independente de regressor foi realizada, de forma similar à anteriormente apresentada, considerando o RRMSE para cada variável resposta. Nesse

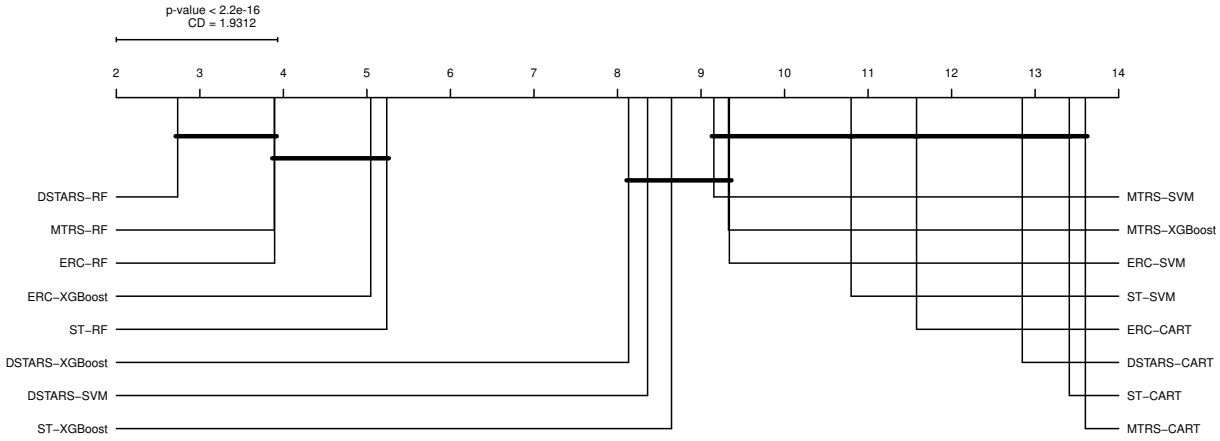

Figura 10 – Resultados do teste de Friedman e do teste *post hoc* de Nemenyi com $\alpha = 0.05$ para os valores de RRMSE de cada *target* nas bases de dados avaliadas, considerando todos métodos MT e regressores

caso, levando em conta as melhores posições no ranqueamento obtidas pelo ERC quando utilizando o XGBoost e a CART, a ordem aparição dos métodos MT se inverteu no primeiro grupo. Todavia, não foi observada diferença estatística entre o ERC e o DSTARS. Nessa análise o MTRS foi significantemente melhor que o ST.

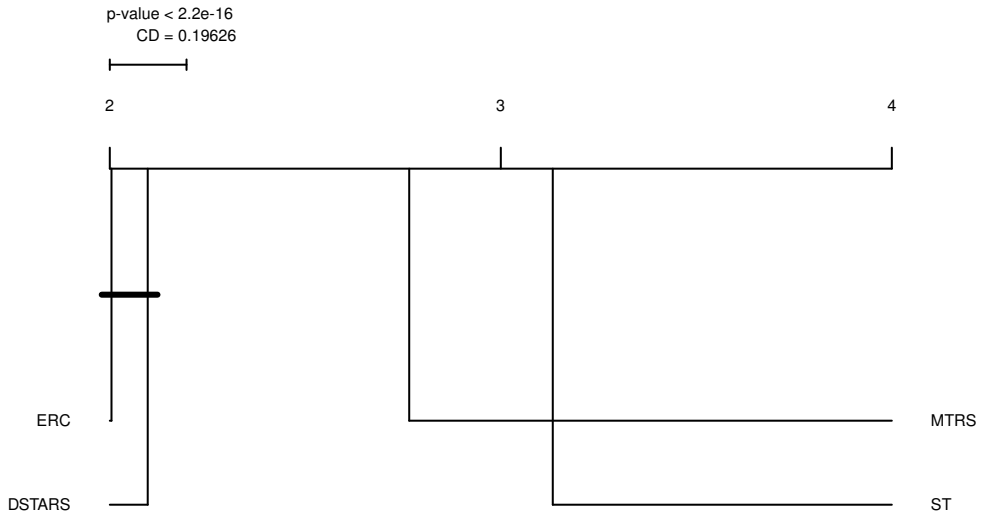

Figura 11 – Resultados do teste de Friedman e do teste *post hoc* de Nemenyi com $\alpha = 0.05$ para os valores de RRMSE de cada *target* nas bases de dados avaliadas, considerando todos métodos MT, independentemente do regressor utilizado

Considerando ainda que, como apresentado na Seção 6.1, o regressor RF obteve os menores erros preditivos na maior parte dos casos, independentemente do método MT empregado, análises estatísticas complementares considerando apenas essa técnica de regressão foram realizadas. Foram contabilizados tanto o aRRMSE obtido para cada base de dados, quanto os valores de RRMSE de cada variável alvo.

Primeiramente, a Figura 12 apresenta os resultados obtidos considerando o aRRMSE de todas as abordagens MT em todos as bases de dados avaliadas, utilizando ape-

nas a RF como regressor. É possível perceber uma grande margem de separação, ainda que não estatisticamente significante, entre o DSTARS e o ERC. O método proposto foi estatisticamente melhor que o MTRS e ST, que por outro lado ficaram ligados ao ERC por um segundo grupo não significantemente diferente.

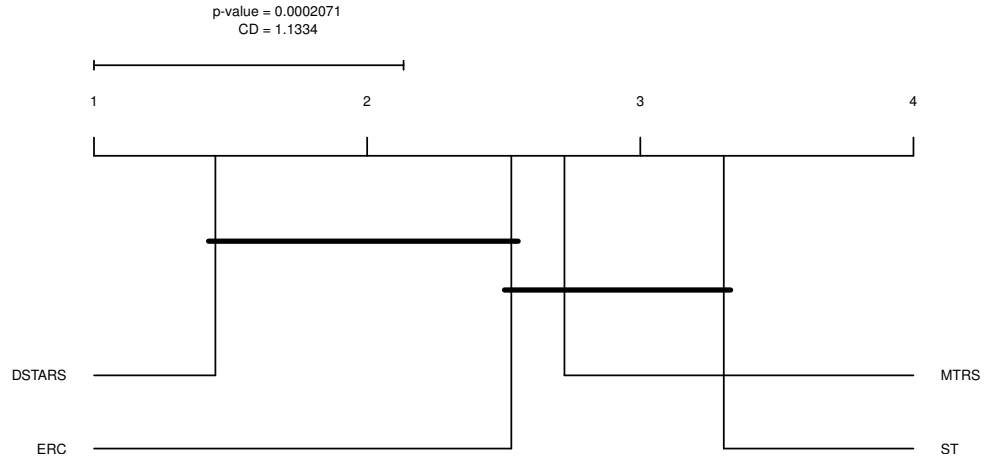

Figura 12 – Resultados do teste de Friedman e do teste *post hoc* de Nemenyi com $\alpha = 0.05$ para os valores de aRRMSE em cada base de dados avaliada, considerando todos métodos MT e o regressor RF

Por fim, uma análise considerando o RRMSE de cada *target* foi realizada utilizando o apenas o regressor RF. Muitas vezes o uso da média entre os RRMSEs, o que resulta no aRRMSE, pode mascarar a proeminência ou inferioridade de determinada abordagem MT, devido à atenuação da possível variabilidade existente durante a agregação das respostas. Por essa razão, uma análise *target*-a-*target* é relevante.

A Figura 13 apresenta os resultados obtidos para essa variação de análise. Quando avaliando a eficiência de cada abordagem MT aliada com o regressor que mostrou prover os menores resultados de erro, o DSTARS se mostrou ser estatisticamente melhor que os demais métodos por uma grande margem. Ademais, nessa comparação o MTRS obteve melhores resultados que o ERC, ainda que não tenham sido observadas diferenças significantes entre as duas abordagens. Por fim, a abordagem ST obteve os piores resultados, estando isolada na última posição, o que constitui mais um indicativo dos ganhos trazidos pela inserção das dependências entre as variáveis alvo na modelagem de problemas de regressão MT.

6.4 DSTARS: análise dos modelos gerados, parâmetros escolhidos e o impacto do *tuning*

Primeiramente, uma análise dos valores de hiper-parametrização obtidos após o procedimento de *tuning*, descrito na Seção 5.3, foi realizada. A Tabela 7 sumariza os valores de ϕ encontrados após a execução do *tuning*, bem como os valores médios de erro

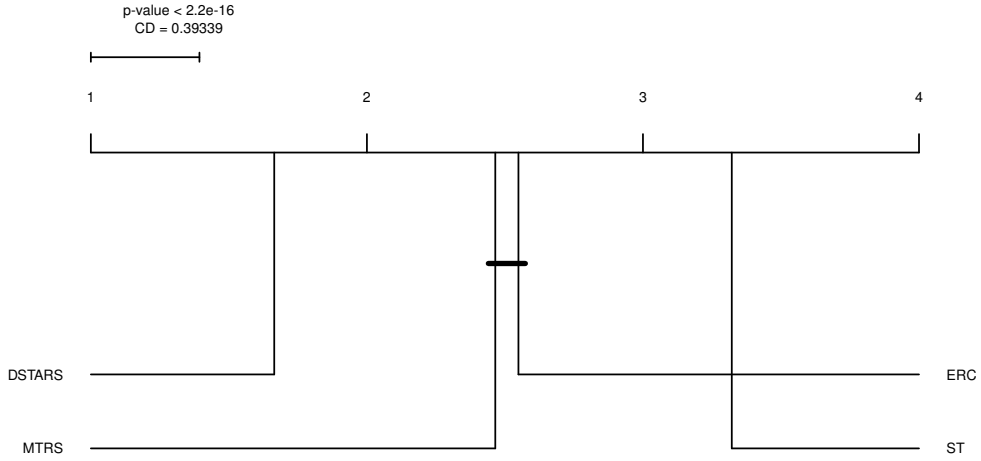

Figura 13 – Resultados do teste de Friedman e do teste *post hoc* de Nemenyi com $\alpha = 0.05$ para os valores de RRMSE de cada *target* nas bases de dados avaliadas, considerando todos métodos MT e o regressor RF

obtidos. Os valores de ϕ mais à esquerda correspondem às parametrizações que geraram os menores valores de aRRMSE, enquanto os valores reportados entre parênteses indicam os valores do hiper-parâmetro que levaram aos valores medianos de erro. Em adição, nas colunas correspondentes ao aRRMSE, os valores médios e o desvio padrão obtidos para essa métrica são reportados, considerando todos os valores de ϕ avaliados.

Tabela 7 – Valores de ϕ que geraram os menores valores de erro durante o procedimento de *tuning* e, entre parênteses, os valores de ϕ que originaram os valores medianos de aRRMSE. Os valores médios de aRRMSE obtidos dentre todos os valores ϕ são reportados juntamente com o desvio padrão observado

Base	RF		SVM		XGBoost		CART	
	ϕ	aRRMSE	ϕ	aRRMSE	ϕ	aRRMSE	ϕ	aRRMSE
atp1d	0.5 (0.8)	0.3910 ± 0.0012	0.6 (0.2)	0.4400 ± 0.0003	0.6 (0.9)	0.40659 ± 0.00302	0.6 (0.7)	0.4757 ± 0.0060
atp7d	0.1 (0.7)	0.5187 ± 0.0015	0.7 (0.1)	0.6411 ± 0.0006	0.5 (0.7)	0.52124 ± 0.02513	0.0 (0.9)	0.6977 ± 0.0041
oes97	0.6 (0.5)	0.5154 ± 0.0014	0.3 (0.7)	0.6117 ± 0.0006	0.6 (0.5)	0.58450 ± 0.00663	0.0 (0.6)	0.6958 ± 0.0021
oes10	0.5 (0.0)	0.4079 ± 0.0014	0.2 (0.6)	0.5458 ± 0.0003	0.9 (0.5)	0.47489 ± 0.00722	0.3 (0.5)	0.5987 ± 0.0023
rf1	0.0 (0.5)	0.0615 ± 0.0068	0.0 (0.5)	0.1022 ± 0.0023	0.2 (0.5)	0.12693 ± 0.00278	0.1 (1.0)	0.3561 ± 0.0065
rf2	0.0 (0.5)	0.0819 ± 0.0037	0.0 (0.5)	0.1068 ± 0.0004	0.5 (0.7)	0.12912 ± 0.00166	0.7 (0.5)	0.3643 ± 0.0058
scm1d	0.2 (0.5)	0.2731 ± 0.0014	0.1 (0.5)	0.3220 ± 0.0011	0.3 (0.7)	0.30614 ± 0.00090	0.0 (0.5)	0.5020 ± 0.0025
scm20d	0.2 (0.5)	0.3159 ± 0.0040	0.1 (0.5)	0.3337 ± 0.0033	0.3 (0.3)	0.36258 ± 0.00026	0.0 (0.5)	0.7074 ± 0.0050
edm	0.5 (0.9)	0.6738 ± 0.0119	0.6 (0.5)	0.7771 ± 0.0088	0.6 (0.2)	0.69129 ± 0.01508	0.7 (0.5)	0.7240 ± 0.0229
sf1	0.8 (0.5)	1.0329 ± 0.0504	0.9 (0.2)	0.9539 ± 0.0211	0.0 (0.2)	1.35014 ± 0.00935	0.6 (0.5)	0.9889 ± 0.0338
sf2	0.8 (0.6)	0.8691 ± 0.0368	0.2 (0.1)	0.7806 ± 0.0041	0.6 (0.5)	1.08265 ± 0.02302	0.5 (0.6)	0.8317 ± 0.0015
jura	0.4 (0.1)	0.5993 ± 0.0050	0.6 (0.3)	0.6451 ± 0.0073	0.9 (0.5)	0.64031 ± 0.00175	0.7 (0.2)	0.7143 ± 0.0105
wq	0.8 (0.5)	0.9168 ± 0.0175	0.5 (0.3)	0.9723 ± 0.0343	0.5 (0.5)	0.99026 ± 0.01378	0.6 (0.5)	0.9900 ± 0.0176
enb	0.5 (0.8)	0.1199 ± 0.0023	0.3 (0.1)	0.1699 ± 0.0025	0.6 (0.3)	0.06071 ± 0.00100	0.6 (0.5)	0.2938 ± 0.0036
slump	0.8 (1.0)	0.8372 ± 0.0136	0.5 (0.6)	0.7077 ± 0.0287	0.6 (0.5)	0.85544 ± 0.00436	0.7 (0.5)	1.0445 ± 0.0186
andro	0.3 (0.8)	0.7284 ± 0.0422	0.0 (0.5)	0.8628 ± 0.1122	0.5 (0.7)	0.47473 ± 0.00845	0.3 (0.0)	0.6467 ± 0.0182
osales	0.3 (0.8)	0.7453 ± 0.0178	0.0 (0.5)	1.1716 ± 0.0012	0.9 (0.5)	0.85389 ± 0.02249	0.2 (0.1)	0.9511 ± 0.0062
scfp	0.6 (0.8)	0.9314 ± 0.0242	0.9 (0.6)	0.8323 ± 0.0068	0.5 (0.6)	1.13035 ± 0.01740	0.1 (0.2)	1.0836 ± 0.0331

A partir dos valores obtidos de ϕ não fica clara a possibilidade de se traçar um paralelo entre as características da base de dados avaliada e um valor mais adequado para o hiper-parâmetro. Em casos específicos, como os observados para os conjuntos *rf1* e *scm20d*, todos os regressores obtiveram os menores valores de erro quando utilizando

valores baixos de ϕ. Inclusive, a melhor hiper-parametrização encontrada para as bases de dados *scm1d* e *scm20d* foi a mesma independentemente do regressor, o que decorre da similaridade existente entre os conjuntos. No entanto, em grande parte dos casos o mesmo padrão não se repetiu, como por exemplo, nas bases *atp7d* e *sf1*. A mesma análise pode ser estendida para os valores medianos de ϕ observados. De fato, em alguns casos, como na base de dados *wq*, os valores medianos de aRRMSE observados dentre todos os modelos DSTARS induzidos durante o *tuning*, foram obtidos pela escolha de $\phi = 0.5$. Outros casos parecidos podem ser verificados nos conjuntos *oes97*, *rf1*, *rf2*, *scm1d* e *scm20d*, entre outros. No entanto, houveram também casos onde os valores de ϕ correspondentes aos índices medianos de erro preditivo diferiram dependendo do regressor escolhido (vide as bases de dados *andro*, *edm* e *jura*, por exemplo).

A inexistência de um padrão claro para os melhores valores de ϕ e as diferenças observadas pela utilização de diferentes regressores motivaram análises adicionais do impacto do *tuning* no desempenho final do DSTARS. Em um cenário otimista, o desempenho final não seria grandemente influenciado pelo valor de ϕ empregado, o que levaria a valores baixos de desvio padrão de aRRMSE em todos os casos. No entanto, como reportado na Tabela 7, os valores de aRRMSE variaram em diferentes níveis dependendo do regressor escolhido, mesmo quando se tratando do mesmo problema de regressão MT avaliado. Dessa forma, as características dos regressores utilizados são fatores a serem levados em conta para a escolha de ϕ. Como forma de oferecer informações complementares àquelas apresentadas na Tabela 7, considerando os diferentes valores de aRRMSE obtidos para cada ϕ, gráficos de caixa (*boxplots*) foram empregados. A amplitude das caixas é um indicativo de estabilidade do método em relação à parametrização: caixas amplas implicam em grandes variações do aRRMSE para diferentes valores ϕ; em contrapartida, caixas estreitas são uma indicação de pouca sensibilidade no valor de erro em relação ao parâmetro do DSTARS escolhido.

Primeiramente o regressor CART foi considerado, tendo em vista que se trata do modelo mais simples avaliado e, consequentemente, poderia estar mais sujeito a depender dos valores de parametrização empregados. A Figura 14 apresenta um *boxplot* dos valores de aRRMSE obtidos pelo DSTARS com o regressor CART, considerando todos os valores de ϕ avaliados. Como previamente sugerido, é possível perceber grandes variações nos erros obtidos em muitas das bases. Em casos pontuais, como nas bases *atp1d* e *jura*, a escolha do valor de ϕ não trouxe grandes impactos no erro final computado, todavia, no conjunto *andro*, por exemplo, uma escolha errônea do parâmetro do DSTARS poderia tornar o desempenho do método MT inferior a se predizer constantemente a média de cada alvo, fato indicado pela obtenção de um aRRMSE superior a 1.

A mesma análise é apresentada para o regressor XGBoost, como apresentado na Figura 15. Para esse regressor cenários extremos são observáveis: enquanto que em alguns

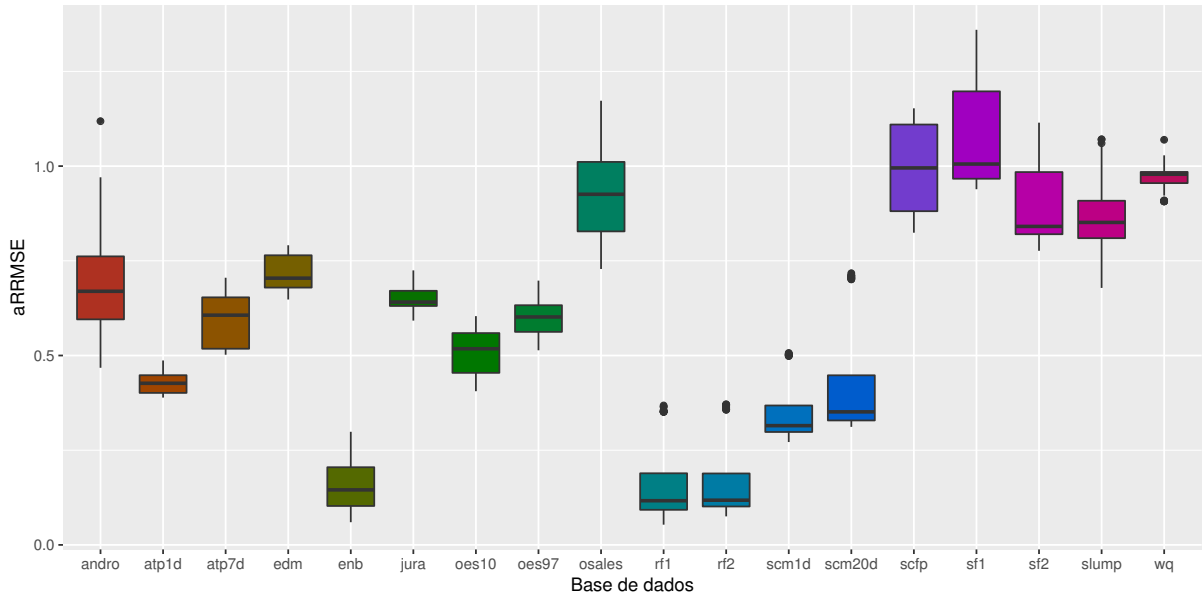

Figura 14 – Análise da variação dos valores de aRRMSE obtidos pelo DSTARS para diferentes valores de ϕ e o regressor CART

casos a escolha do valor de ϕ não impactou de forma relevante no aRRMSE obtido (*jura*, *rf1* e *rf2*, por exemplo), em outros foram observadas grandes variabilidades dos resultados (*osales* e *scfp*, por exemplo). Assim, pode-se afirmar que em determinadas situações o XGBoost é mais sensível aos valores de ϕ escolhidos do que a própria CART. Os resultados obtidos vão de encontro com a análise realizada previamente, relativa a combinação do regressor XGBoost com o ERC.

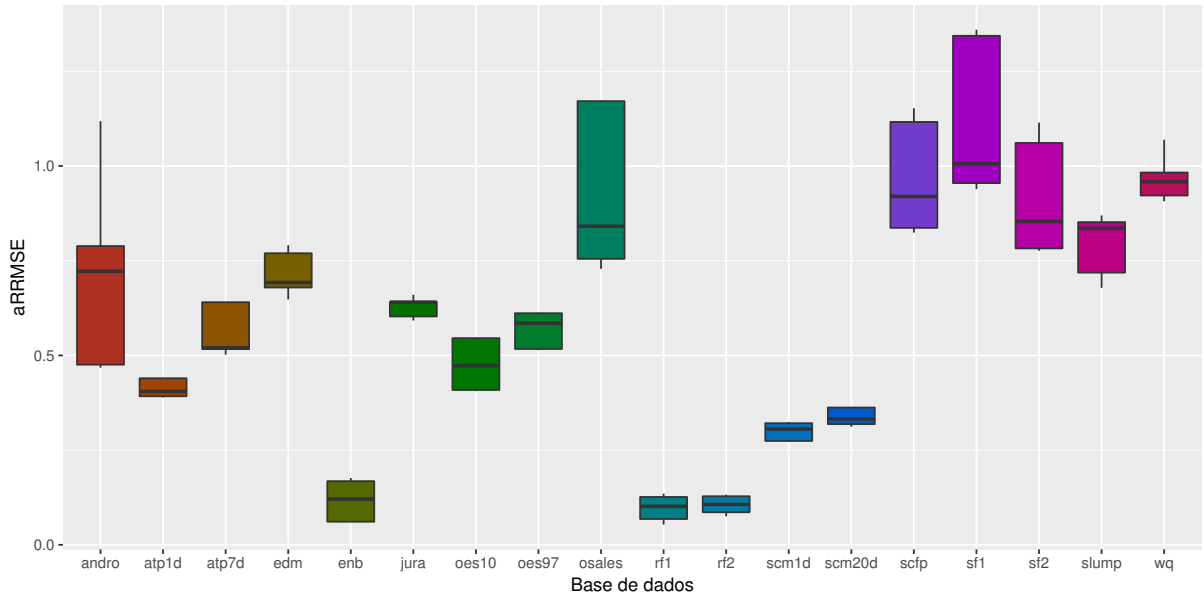

Figura 15 – Análise da variação dos valores de aRRMSE obtidos pelo DSTARS para diferentes valores de ϕ e o regressor XGBoost

De forma similar, a SVM demonstrou grandes variações no erro preditivo a partir

da variação dos valores de ϕ do DSTARS, como apresentado na Figura 16. No entanto, exceto para a base de dados *osales*, as variações de aRRMSE observadas foram menores do que as do XGBoost. De fato, é de conhecimento que a SVM, apesar de ser muito flexível e apresentar grande poder preditivo, pode ser grandemente afetada pelos seus valores de parametrização [40], devendo ser ajustada para cada problema tratado. O fato de todos os regressores terem sido mantidos com seus valores padrão de parametrização pode explicar a variabilidade observada para a SVM.

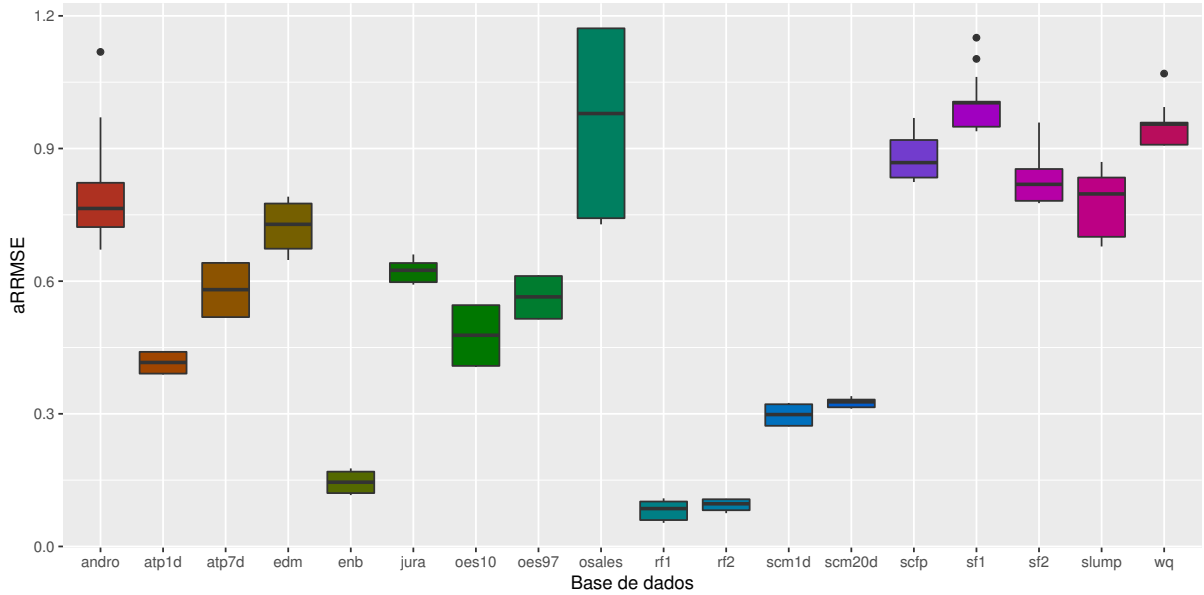

Figura 16 – Análise da variação dos valores de aRRMSE obtidos pelo DSTARS para diferentes valores de ϕ e o regressor SVM

Por fim, a análise de variância do aRRMSE em relação ao ϕ escolhido com a RF é apresentada na Figura 17. A combinação desse regressor com o DSTARS resultou em pequenas variabilidades nos valores de erro para praticamente todos os casos. Esse fato é justificável pela natureza do preditor empregado. De fato, a RF a partir da utilização da amostragem *boostrap*, utilizada na agregação dos preditores através do *bagging*, combinada com a subamostragem de atributos de entrada utilizados para a construção de cada árvore de regressão, é mais robusta à presença de *outliers* e ruídos nas bases de dados. Dessa forma, esse preditor é capaz de se adaptar a múltiplos cenários diferentes [25, 23], mesmo sem o refinamento de sua parametrização. Assim, de acordo com os resultados obtidos, o uso de um valor fixo e possivelmente baixo de ϕ, quando aliado com a RF, não resultaria em relevante degradação no desempenho preditivo do DSTARS.

A partir das análises realizadas, é seguro afirmar que o impacto do valor de ϕ para o DSTARS está diretamente relacionado a escolha do regressor empregado, e capacidade deste a se adaptar a diferentes cenários. Ademais, diferentes valores do limiar ϕ impactam diretamente no número de camadas utilizadas pelo DSTARS para modelar cada problema. O número de camadas utilizadas para cada base de dados, quando utilizando os melhores

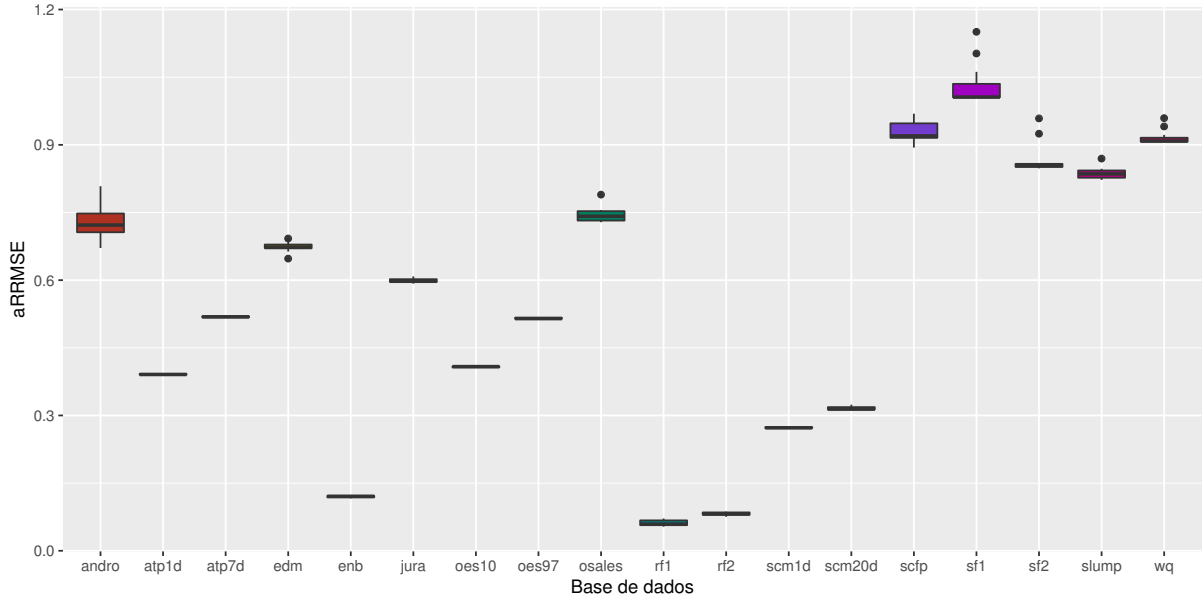

Figura 17 – Análise da variação dos valores de aRRMSE obtidos pelo DSTARS para diferentes valores de ϕ e o regressor RF

valores de ϕ encontrados, são apresentados na Tabela 8.

Tabela 8 – Número médio de camadas de regressores utilizadas pelo DSTARS para cada base de dados e regressores. Todas as variáveis alvo e dobras da validação cruzada foram consideradas

Base	RF	SVM	XGBoost	CART
atp1d	2.2	1.0	1.1	1.1
atp7d	8.0	1.0	1.5	4.6
oes97	1.1	1.5	1.1	3.0
oes10	1.7	1.7	1.0	1.0
rf1	6.7	9.0	2.2	2.0
rf2	6.7	4.0	1.7	1.2
scm1d	9.5	4.1	1.2	5.3
scm20d	20.6	6.4	2.0	6.6
edm	1.9	1.3	1.0	1.0
sf1	1.0	1.0	3.5	1.0
sf2	1.0	3.0	1.0	1.1
jura	3.1	1.0	1.0	1.0
wq	1.0	2.2	1.0	1.0
enb	2.4	5.9	1.1	1.0
slump	1.3	1.4	1.1	1.1
andro	3.8	13.5	1.7	1.6
osales	4.2	2.8	1.0	3.7
scfp	1.6	1.0	1.1	3.1

Ao contrário do esperado, na grande maioria dos casos o número de camadas de regressores utilizados pelo DSTARS foi relativamente pequeno. Isso vai contra a premissa de utilização de um número elevado de camadas que se tinha inicialmente. Logicamente,

os resultados obtidos estão atrelados à escolha do valor de ε e são campo para futura pesquisa. No entanto, a adição de um regressor que em média diminua o erro preditivo em menos de 10^{-4} (referente ao valor de ε escolhido) não parece ser uma boa escolha, levando em conta o custo computacional adicional envolvido.

Apesar disso, é interessante notar que os regressores que mais apresentaram casos onde o número de camadas foi mais elevado (na base *scm20d*, por exemplo) foram a RF e a SVM. Essas mesmas técnicas de regressão obtiveram as menores taxas de erro quando combinadas com o DSTARS, o que reforça a hipótese de redução de erros com a introdução de múltiplas camadas de regressores. Para informações completas acerca do número médio de camadas empregados pelo DSTARS para cada *target* em todas as bases de dados de *benchmark* avaliadas, o leitor pode consultar a Tabela 9 presente no Apêndice A.

6.5 Comparação dos custos computacionais das soluções para regressão MT

Analisando as escolhas feitas durante os experimentos realizados e, levando em consideração as escolhas feitas durante a execução dos experimentos, uma análise comparativa de complexidade de tempo do DSTARS em relação aos outros métodos de regressão MT será apresentada. Como já apresentado na No Capítulo 3, a complexidade assintótica de tempo da estratégia ST é $O(d\,b)$, sendo d o número de alvos do problema tratado e b o custo do regressor empregado. Consequentemente, o custo do MTRS é $O(2\,d\,b)$, já que exatamente o dobro de regressores são induzidos, em relação à estratégia ST. O ERC, como já descrito na Seção 3.2.2, possui o custo assintótico de $O(10\,d\,b)$, devido à restrição imposta pelos autores do método no processo combinatório de exploração de diferentes ordenações entre os *targets*. O DSTARS, por sua vez, apresenta um custo de $O(d\,[r + (P + 1)\,L\,b])$, onde r representa o custo da métrica de importância empregada, P representa o número de partições utilizadas nas etapas de *Filtering* e *Tracking*, e L o número máximo de camadas de regressores empregadas. Levando em consideração que uma estratégia de validação cruzada com 10 dobras foi empregada para particionar os conjuntos de entrada, o custo do DSTARS nos experimentos realizados foi $O(d\,[r + 11\,L\,b])$.

Considerando que nos experimentos a RF se mostrou a técnica de regressão mais efetiva, sendo esta também empregada para o cálculo das correlações entre os alvos, esse algoritmo será empregado para a análise dos custos dos métodos MT. Versões sem poda de algoritmos de árvores de decisão, como a CART, apresentam uma complexidade de tempo de $O(v\,N\,\log N)$, onde v representa o número de atributos descritivos, e N o número de instâncias de treinamento do problema tratado [30]. Na RF, t árvores CART são empregadas para a construção do *ensemble*, todavia, a cada ponto de corte durante o

treinamento, apenas s atributos aleatoriamente selecionados são considerados, sendo que comumente $s << v$. No algoritmo da RF, t e s são hiper-parâmetros ajustáveis. Assim, o custo de treinamento da RF é $O(t\,s\,N\,\log N)$.

Portanto, o custo total da estratégia ST, considerando a RF como regressor, é $O(d\,t\,s\,N\,\log N)$. Similarmente, os custos dos métodos MTRS e ERC são, respectivamente, $O(2\,d\,t\,s\,N\,\log N)$ e $O(10\,d\,t\,s\,N\,\log N)$. Considerando que a RF foi empregada para mensurar as correlações entre as saídas, por simplicidade, é possível assumir que $r = b$, apesar dos regressores treinados considerarem d atributos de treinamento (correspondentes às aproximações dos *targets*, descritas na Seção 4.3), ao invés de v. Assim, custo total do DSTARS no cenário considerado é $O(d\,[b + 11\,L\,b])$, ou de forma simplificada, $O((1 + 11\,L)\,d\,b)$. Inserindo o custo da RF na expressão, obtém-se o custo final de $O((1 + 11\,L)\,d\;t\,s\,N\,\log N)$ para o DSTARS.

De fato, o DSTARS é o método mais custoso dentre os avaliados, levando em conta os procedimentos que realiza internamente e as decisões empregadas para particionamento do dados. No entanto, a superioridade significativa do DSTARS em relação aos outros métodos, considerando o erro preditivo, faz com que a solução proposta seja a melhor escolha de forma geral. Além disso, o custo mencionado é relativo à indução dos modelos preditivos que, uma vez treinados, poderão ser utilizados com um custo reduzido. A partir do momento que nos experimentos realizados a RF demonstrou pouca variabilidade no erro preditivo, considerando o hiper-parâmetro ϕ do DSTARS, uma estratégia mais leve de particionamento poderia ser empregada para definição do melhor número de camadas de regressores. Essa escolha influenciaria no termo multiplicador da variável L, presente na expressão de complexidade assintótica do DSTARS.

A partir da utilização de uma estratégia de amostragem como a *boostrap* (sem repetição), o custo seria reduzido amplamente, eliminando a necessidade de aplicação do limiar ϕ, potencialmente sem degradar o desempenho preditivo de forma significativa. No contexto hipotético mencionado, a complexidade de treinamento do DSTARS se tornaria $O((1 + 2\,L)\,d\;t\,s\,N\,\log N)$, o que se aproxima da complexidade assintótica das outras abordagens, podendo fazer com que o método proposto seja, inclusive, menos custoso do que o ERC, dependendo do valor de L.

7 CONCLUSÕES E PESQUISAS FUTURAS

Neste trabalho uma nova técnica para regressão multi-target, denominada DSTARS, foi proposta. A solução proposta está pautada na transformação do espaço do problema, sem a necessidade de adaptação de algoritmos de regressão. O método DSTARS se baseia no uso de sucessivas camadas de regressores ST de forma a minimizar o erro de cada variável alvo de forma assíncrona. O modelo final de predição gerado compreende um número diferente de regressores para cada *target*, sendo que as camadas adotadas para cada saída não são necessariamente contíguas. Além disso, o método proposto leva em consideração a dependência estatística existente entre *targets* através da métrica de importância de variáveis da Random Forest. Todavia, a estimativa utilizada é mais realística por comparar como predições de modelos ST se relacionam com os valores verdadeiros das saídas.

O desempenho do DSTARS foi avaliado em 18 bases de dados de *benchmarking*, já reportadas na literatura anteriormente, comparando seu desempenho preditivo com a abordagem ST e dois métodos MT (MTRS e ERC). Quatro algoritmos de regressão (SVM, RF, XGBoost e CART) foram utilizados como regressores base para todas abordagens testadas, sem terem seus valores originais de parametrização alterados.

Os resultados demostraram que o DSTARS gerou menores valores de erro preditivo na maioria dos casos. Quando utilizados todos os regressores o DSTARS se mostrou favoravelmente comparável, mas não estatisticamente superior às abordagens MT comparadas. No entanto, ao utilizar o regressor que resultou nos menores erros preditivos na maioria dos casos (RF), o DSTARS foi estatisticamente superior às outras abordagens.

Além disso, a escolha do hiperparâmetro ϕ do DSTARS se mostrou um fator impactante no desempenho apenas quando utilizando a SVM, XGBoost e CART. Esses regressores apresentam maior variabilidade nos erros obtidos dependendo de como foram configurados. A RF, por sua vez, demonstrou ter maior capacidade de adaptação a diferentes cenários, fato demonstrado com a baixa variação no erro preditivo a partir do uso de diferentes valores de ϕ. Em adição, o número médio de camadas utilizado pelo DSTARS quando combinado ao XGBoost e CART frequentemente foi baixo, estando muito próximo ao que seria equivalente às abordagens ST e MTRS. Combinado com a RF e SVM, o DSTARS apresentou mais casos onde o número de camadas de regressores empregados foi maior. Vale mencionar que nos casos onde o número de camadas empregadas foi pequeno, o desempenho do DSTARS ainda foi superior ou comparável às abordagens ST e MTRS.

Como próximos passos, problemas da vida real deverão ser testados, comparando o desempenho do método proposto contra outras abordagens MT e, avaliando o impacto da escolha dos hiper-parâmetros do DSTARS nos resultados obtidos. Em adição, uma análise

de valores de parametrização mais adequados para os regressores base pode ser realizada, levando em consideração o aumento da dimensão dos problemas que é trazido pelo uso das técnicas MT (MTRS, ERC e DSTARS) e, tornando assim mais justa a comparação dessas abordagens com a variante ST.

Por fim, espera-se avaliar outras estratégias de amostragem para o procedimento de *Tracking* do DSTARS, visando diminuir o custo computacional dessa etapa. O emprego de uma abordagem mais leve, como a amostragem *bootstrap*, em detrimento da validação cruzada utilizada atualmente, poderia diminuir o custo da solução e eliminar a necessidade do uso do parâmetro ϕ para a escolha das camadas que repetidamente trouxeram ganhos preditivos nos diferentes conjuntos de validação atualmente avaliados na etapa de *Tracking*.

REFERÊNCIAS

[1] MITCHELL, T. *Machine Learning*. McGraw-Hill Education, 1997. (McGraw-Hill international editions - computer science series). ISBN 9780070428072. Disponível em: <https://books.google.com.br/books?id=xOGAngEACAAJ>.

[2] BRAGA, A. d. P.; LUDERMIR, T.; CARVALHO, A. C. de. *Redes Neurais: Teoria e Aplicações*. [S.l.]: LTC-Livros Técnicos e Científicos Editora SA, Rio, 2000.

[3] HADAVANDI, E.; SHAHRABI, J.; SHAMSHIRBAND, S. A novel Boosted-neural network ensemble for modeling multi-target regression problems. *Eng. Appl. Artif. Intell.*, Elsevier, v. 45, p. 204–219, 2015. ISSN 09521976. Disponível em: <http://dx.doi.org/10.1016/j.engappai.2015.06.022>.

[4] BORCHANI, H. et al. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 5, n. 5, p. 216–233, 2015.

[5] SPYROMITROS-XIOUFIS, E. et al. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, Springer, v. 104, n. 1, p. 55–98, 2016.

[6] CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2016. (KDD '16), p. 785–794. ISBN 978-1-4503-4232-2. Disponível em: <http://doi.acm.org/10.1145/2939672.2939785>.

[7] KOCEV, D. et al. Using single- and multi-target regression trees and ensembles to model a compound index of vegetation condition. *Ecological Modelling*, Elsevier, v. 220, n. 8, p. 1159–1168, 2009.

[8] MELKI, G. et al. Multi-target support vector regression via correlation regressor chains. *Information Sciences*, Elsevier, v. 415, p. 53–69, 2017.

[9] AHO, T. et al. Multi-Target Regression with Rule Ensembles. *J. Mach. Learn. Res.*, v. 13, p. 2367–2407, 2012. ISSN 1532-4435.

[10] KOCEV, D. et al. Ensembles of multi-objective decision trees. In: *Proceedings of the 18th European Conference on Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 2007. (ECML '07), p. 624–631. ISBN 978-3-540-74957-8. Disponível em: <http://dx.doi.org/10.1007/978-3-540-74958-5_61>.

[11] XU, S. et al. Multi-output least-squares support vector regression machines. *Pattern Recognit. Lett.*, Elsevier B.V., v. 34, n. 9, p. 1078–1084, 2013. ISSN 01678655. Disponível em: <http://dx.doi.org/10.1016/j.patrec.2013.01.015>.

[12] ZHANG, S. et al. Leverage triple relational structures via low-rank feature reduction for multi-output regression. *Multimed. Tools Appl.*, Multimedia Tools and Applications, 2016. ISSN 1380-7501. Disponível em: <http://dx.doi.org/10.1007/s11042-016-3980-3>.

[13] KOCEV, D.; CECI, M. Ensembles of extremely randomized trees for multi-target regression. In: *Discovery Science*. Springer International Publishing, 2015. p. 86–100. Disponível em: <https://doi.org/10.1007/978-3-319-24282-8_9>.

[14] OSOJNIK, A.; DŽEROSKI, S.; KOCEV, D. Option predictive clustering trees for multi-target regression. In: *Discovery Science*. Springer International Publishing, 2016. p. 118–133. Disponível em: <https://doi.org/10.1007/978-3-319-46307-0_8>.

[15] MOYANO, J. M.; GIBAJA, E. L.; VENTURA, S. An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains. In: *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/cec.2017.7969548>.

[16] SANTANA, E. J.; MASTELINI, S. M.; Barbon Jr., S. Deep Regressor Stacking for Air Ticket Prices Prediction. In: *XIII Brazilian Symposium on Information Systems: Information Systems for Participatory Digital Governance*. Brazilian Computer Society (SBC), 2017. p. 25–31. Disponível em: <http://sbsi2017.dcc.ufla.br/download/proceedings_completo.pdf>.

[17] MASTELINI, S. M. et al. DSTARS: A multi-target deep structure for tracking asynchronous regressor stack. In: *2017 Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/bracis.2017.30>.

[18] FIORE, U. et al. Network anomaly detection with the restricted boltzmann machine. *Neurocomputing*, Elsevier, v. 122, p. 13–23, 2013.

[19] HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning*. Springer New York, 2009. Disponível em: <https://doi.org/10.1007/978-0-387-84858-7>.

[20] ZHU, X. Semi-supervised learning literature survey. Citeseer, 2005.

[21] KRISHNAPURAM, B. *Cost-sensitive machine learning*. Boca Raton, FL: CRC Press, 2012. ISBN 978-1439839256.

[22] KOCEV, D. et al. Tree ensembles for predicting structured outputs. *Pattern Recognition*, Elsevier, v. 46, n. 3, p. 817–833, 2013.

[23] FERNÁNDEZ-DELGADO, M. et al. Do we need hundreds of classifiers to solve real world classification problems. *J. Mach. Learn. Res*, v. 15, n. 1, p. 3133–3181, 2014.

[24] KOTSIANTIS, S. B.; ZAHARAKIS, I.; PINTELAS, P. *Supervised machine learning: A review of classification techniques*. 2007.

[25] BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

[26] VAPNIK, V. N. *The Nature of Statistical Learning Theory*. Springer New York, 2000. Disponível em: <https://doi.org/10.1007/978-1-4757-3264-1>.

[27] DEKA, P. C. et al. Support vector machine applications in the field of hydrology: a review. *Applied soft computing*, Elsevier, v. 19, p. 372–386, 2014.

[28] PAPADOPOULOU, O. S. et al. Sensory and microbiological quality assessment of beef fillets using a portable electronic nose in tandem with support vector machine analysis. *Food Research International*, v. 50, n. 1, p. 241 – 249, 2013. ISSN 0963-9969.

[29] LOH, W.-Y. Classification and regression trees. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley Online Library, v. 1, n. 1, p. 14–23, 2011.

[30] BREIMAN, L. et al. *Classification and Regression Trees*. Taylor & Francis, 1984. (The Wadsworth and Brooks-Cole statistics-probability series). ISBN 9780412048418. Disponível em: <https://books.google.com.br/books?id=JwQx-WOmSyQC>.

[31] TSOUMAKAS, G. et al. Multi-target regression via random linear target combinations. In: *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2014. p. 225–240. Disponível em: <https://doi.org/10.1007/978-3-662-44845-8_15>.

[32] CROXTON, F. E.; COWDEN, D. J. Applied general statistics. New York, NY, US: Prentice-Hall, Inc, 1939.

[33] GAMA, J.; BRAZDIL, P. Cascade generalization. *Machine Learning*, v. 41, n. 3, p. 315–343, Dec 2000. ISSN 1573-0565. Disponível em: <http://dx.doi.org/10.1023/A:1007652114878>.

[34] LIU, G.; LIN, Z.; YU, Y. Multi-output regression on the output manifold. *Pattern Recognit.*, Elsevier, v. 42, n. 11, p. 2737–2743, 2009. ISSN 00313203. Disponível em: <http://dx.doi.org/10.1016/j.patcog.2009.05.001>.

[35] ZHANG, W. et al. Multi-output LS-SVR machine in extended feature space. *CIMSA 2012 - 2012 IEEE Int. Conf. Comput. Intell. Meas. Syst. Appl. Proc.*, p. 130–144, 2012.

[36] XIONG, T.; BAO, Y.; HU, Z. Multiple-output support vector regression with a firefly algorithm for interval-valued stock price index forecasting. *Knowledge-Based Systems*, Elsevier, v. 55, p. 87–100, 2014.

[37] BLOCKEEL, H.; RAEDT, L. D. Top-down induction of first-order logical decision trees. *Artificial intelligence*, Elsevier, v. 101, n. 1-2, p. 285–297, 1998.

[38] DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, JMLR. org, v. 7, p. 1–30, 2006.

[39] NATEKIN, A.; KNOLL, A. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, Frontiers Media SA, v. 7, 2013.

[40] MANTOVANI, R. G. et al. To tune or not to tune: Recommending when to adjust SVM hyper-parameters via meta-learning. In: *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015. Disponível em: <https://doi.org/10.1109/ijcnn.2015.7280644>.

Apêndices

APÊNDICE A – RESULTADOS DETALHADOS DOS EXPERIMENTOS REALIZADOS

Tabela 9 – Valores médios de camadas de regressores utilizadas pelo DSTARS levando em consideração os melhores valores de ϕ escolhidos na etapa de *tuning*. Todos os regressores e dobras da validação cruzada foram considerados

Base Target	RF	SVM	XGBoost	CART
atp1d	2.2	1.0	1.1	1.1
ALLminpA	1.6	1.0	1.2	1.1
ALLminp0	2.1	1.0	1.1	1.1
aDLminpA	2.6	1.0	1.1	1.0
aCOminpA	2.4	1.0	1.0	1.0
aFLminpA	2.0	1.0	1.0	1.1
aUAminpA	2.3	1.0	1.0	1.0
atp7d	8.0	1.0	1.5	4.6
ALLminpA	7.4	1.0	1.8	4.6
ALLminp0	8.2	1.0	1.4	4.6
aDLminpA	7.7	1.0	1.5	4.6
aCOminpA	8.7	1.0	1.2	4.6
aFLminpA	8.6	1.0	1.5	4.6
aUAminpA	7.2	1.0	1.8	4.6
oes97	1.1	1.5	1.1	3.0
58028	1.0	2.0	1.0	3.0
15014	1.1	1.6	1.0	3.0
32511	1.1	1.0	1.1	3.0
15017	1.1	2.0	1.0	3.0
98502	1.0	2.7	1.0	3.0
92965	1.0	1.5	1.0	3.0
32314	1.3	1.9	1.0	3.0
13008	1.1	2.0	1.1	3.0
21114	1.0	2.0	1.0	3.0
85110	1.1	1.0	1.0	3.0
27311	1.0	1.0	1.2	3.0
92965	1.0	1.0	1.2	3.0
65032	1.2	1.0	1.1	3.0
92998	1.0	1.0	1.1	3.0

Base Target	RF	SVM	XGBoost	CART
27108	1.0	1.2	1.1	3.0
53905	1.0	1.0	1.2	3.0
oes10	1.7	1.7	1.0	1.0
513021	2.2	1.4	1.0	1.0
292071	2.6	2.0	1.0	1.1
392021	2.0	1.6	1.0	1.0
151131	1.5	1.0	1.0	1.0
151141	2.2	1.5	1.0	1.0
291069	1.4	1.0	1.0	1.1
119032	1.2	1.9	1.0	1.0
432011	1.9	2.0	1.0	1.0
419022	1.8	2.0	1.0	1.0
292037	1.1	2.0	1.0	1.0
519061	1.8	2.0	1.0	1.0
291051	2.1	2.0	1.0	1.0
172141	1.7	1.6	1.0	1.0
431011	1.3	2.0	1.0	1.0
291127	1.8	2.0	1.0	1.0
412021	1.8	1.9	1.0	1.0
rf1	6.7	9.0	2.2	2.0
CHSI2	6.7	9.0	2.2	2.6
NASI2	6.7	9.0	1.2	1.9
EADM7	6.7	9.0	2.3	2.2
SCLM7	6.7	9.0	2.7	2.0
CLKM7	6.7	9.0	2.6	2.2
VALI2	6.7	9.0	2.3	2.3
NAPM7	6.7	9.0	1.7	2.1
DLDI4	6.7	9.0	2.2	1.0
rf2	6.7	4.0	1.7	1.2
CHSI2	6.7	4.0	1.9	1.2
NASI2	6.7	4.0	1.0	1.0
EADM7	6.7	4.0	2.0	1.9
SCLM7	6.7	4.0	2.0	1.4
CLKM7	6.7	4.0	2.0	1.0
VALI2	6.7	4.0	2.0	1.4
NAPM7	6.7	4.0	1.0	1.0
DLDI4	6.7	4.0	2.0	1.0

Base Target	RF	SVM	XGBoost	CART
scm1d	9.5	4.1	1.2	5.3
LBL	9.7	4.5	1.0	5.3
MTLp2	9.0	4.4	1.0	5.3
MTLp3	8.0	4.2	1.0	5.3
MTLp4	10.5	3.5	1.0	5.3
MTLp5	10.0	4.6	2.0	5.3
MTLp6	9.9	4.2	1.9	5.3
MTLp7	10.5	3.8	2.0	5.3
MTLp8	10.6	4.1	1.8	5.3
MTLp9	8.4	4.3	1.0	5.3
MTLp10	9.2	4.4	1.0	5.3
MTLp11	8.5	4.4	1.0	5.3
MTLp12	9.4	3.8	1.0	5.3
MTLp13	10.2	3.5	1.0	5.3
MTLp14	9.4	4.2	1.0	5.3
MTLp15	10.0	4.0	1.0	5.3
MTLp16	8.8	3.0	1.0	5.3
scm20d	20.6	6.4	2.0	6.6
LBL	19.9	6.3	2.0	6.6
MTLp2A	18.5	6.7	2.0	6.6
MTLp3A	21.3	6.8	2.0	6.6
MTLp4A	21.7	7.1	2.0	6.6
MTLp5A	22.8	7.4	2.0	6.6
MTLp6A	20.4	6.3	2.0	6.6
MTLp7A	20.6	6.5	2.0	6.6
MTLp8A	20.4	6.1	2.0	6.6
MTLp9A	19.5	6.1	2.0	6.6
MTLp10A	20.8	5.7	2.0	6.6
MTLp11A	21.9	6.9	2.0	6.6
MTLp12A	19.8	5.6	2.0	6.6
MTLp13A	20.9	7.0	2.0	6.6
MTLp14A	18.7	5.4	2.0	6.6
MTLp15A	21.0	6.3	2.0	6.6
MTLp16A	21.9	5.9	2.0	6.6
edm	1.9	1.3	1.0	1.0
DFlow	2.4	1.0	1.0	1.0
DGap	1.3	1.6	1.0	1.0

Base Target	RF	SVM	XGBoost	CART
sf1	1.0	1.0	3.5	1.0
c.class	1.0	1.0	3.5	1.0
m.class	1.0	1.0	3.5	1.0
x.class	1.0	1.0	3.5	1.0
sf2	1.0	3.0	1.0	1.1
c.class	1.0	4.5	1.0	1.2
m.class	1.0	1.8	1.0	1.1
x.class	1.0	2.7	1.0	1.1
jura	3.1	1.0	1.0	1.0
Cd	3.2	1.0	1.0	1.0
Co	2.8	1.0	1.0	1.0
Cu	3.3	1.0	1.0	1.0
wq	1.0	2.2	1.0	1.0
25400	1.0	1.9	1.0	1.0
29600	1.0	2.1	1.0	1.0
30400	1.0	2.1	1.0	1.0
33400	1.0	3.4	1.0	1.1
17300	1.0	1.8	1.0	1.0
19400	1.0	1.2	1.0	1.0
34500	1.0	3.0	1.0	1.0
38100	1.0	1.7	1.0	1.0
49700	1.0	2.2	1.0	1.0
50390	1.0	2.7	1.0	1.0
55800	1.0	2.1	1.0	1.0
57500	1.0	2.5	1.0	1.0
59300	1.0	2.4	1.0	1.0
37880	1.0	1.2	1.0	1.0
enb	2.4	5.9	1.1	1.0
Y1	2.8	6.2	1.0	1.0
Y2	2.0	5.6	1.2	1.0
slump	1.3	1.4	1.1	1.1
slump	1.0	2.0	1.2	1.1
flow	1.0	1.2	1.1	1.1
cpr_str	2.0	1.0	1.1	1.2
andro	3.8	13.5	1.7	1.6
Target	4.4	13.5	1.6	1.0
Target_2	3.7	13.5	1.6	1.0

Base Target	RF	SVM	XGBoost	CART
Target_3	3.7	13.5	1.4	2.0
Target_4	3.5	13.5	1.7	2.0
Target_5	3.7	13.5	1.5	1.7
Target_6	3.7	13.5	2.1	2.0
osales	4.2	2.8	1.0	3.7
M1	5.3	2.8	1.0	3.5
M2	5.0	2.8	1.0	4.3
M3	4.9	2.8	1.0	3.8
M4	5.1	2.8	1.0	4.1
M5	4.4	2.8	1.0	4.3
M6	3.4	2.8	1.0	4.1
M7	4.1	2.8	1.0	4.1
M8	3.9	2.8	1.0	3.4
M9	4.0	2.8	1.0	3.4
M10	3.2	2.8	1.0	3.3
M11	3.9	2.8	1.0	3.3
M12	2.6	2.8	1.0	3.0
scfp	1.6	1.0	1.1	3.1
views	1.9	1.0	1.0	3.1
votes	1.1	1.0	1.0	3.5
comments	1.8	1.0	1.4	2.7

Tabela 10 – RRMSEs obtidos por todos métodos MT e regressores avaliados. Os menores valores por regressor estão em negrito e os menores por *target* sublinhados

Base Target	RF				SVM				XGBoost				CART			
	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS
atp1d	0.3920	0.3904	0.3902	**0.3889**	**0.4397**	0.4402	0.4398	**0.4397**	0.4048	0.4080	**0.3830**	0.4042	0.4717	0.4843	**0.4683**	0.4704
ALLminpA	0.4767	0.4788	0.4766	**0.4752**	**0.5435**	0.5444	0.5438	**0.5435**	0.4991	0.5043	**0.4871**	0.4995	0.6704	**0.6387**	0.6634	0.6616
ALLminp0	0.4259	0.4320	0.4273	**0.4247**	**0.4589**	0.4594	0.4592	**0.4589**	0.4907	0.4920	**0.4561**	0.4934	0.5875	0.6110	**0.5709**	0.5873
aDLminpA	0.4139	0.4082	0.4111	**0.4070**	**0.4336**	0.4342	0.4338	**0.4336**	0.4340	0.4292	**0.3852**	0.4274	0.4589	0.5017	**0.4489**	0.4589
aCOminpA	0.3091	0.3097	**0.3090**	0.3092	**0.3619**	0.3627	0.3621	**0.3619**	0.2581	0.2660	**0.2472**	0.2581	0.2918	0.2909	**0.2908**	0.2918
aFLminpA	0.4358	**0.4267**	0.4295	0.4307	**0.4856**	0.4862	0.4857	**0.4856**	0.5272	0.5250	**0.4922**	0.5272	**0.5771**	0.6177	0.5912	0.5786
aUAminpA	0.2905	0.2869	0.2880	**0.2865**	**0.3543**	0.3544	0.3544	**0.3543**	**0.2195**	0.2313	0.2302	**0.2195**	**0.2442**	0.2457	0.2447	**0.2442**
atp7d	**0.5164**	0.5169	0.5178	0.5167	**0.6404**	0.6414	0.6410	**0.6404**	0.5068	0.5228	**0.4751**	0.5020	0.6979	0.6805	**0.6570**	0.6889
ALLminpA	**0.5864**	0.5942	0.5914	0.5914	0.7735	**0.7693**	0.7720	0.7735	0.5305	0.5130	0.5194	**0.5031**	0.7793	0.7683	**0.7141**	0.8060
ALLminp0	0.6199	0.6151	0.6201	**0.6079**	**0.6906**	0.6920	0.6917	**0.6906**	0.7053	0.7651	**0.6887**	0.6974	1.0222	0.9967	**0.9961**	1.0253
aDLminpA	0.5130	0.5036	0.5059	**0.5019**	**0.5833**	0.5835	0.5837	**0.5833**	0.5195	0.5491	**0.4719**	0.5274	0.6393	0.6686	**0.6127**	0.7055
aCOminpA	0.3931	**0.3869**	0.3936	0.3973	**0.4988**	0.5029	0.5008	**0.4988**	0.3398	0.3444	**0.3084**	0.3423	0.4445	0.4094	**0.3831**	0.4130
aFLminpA	**0.6108**	0.6246	0.6185	0.6254	**0.7953**	0.7966	0.7955	**0.7953**	0.6572	0.6739	**0.5562**	0.6537	0.9099	0.8567	0.8735	**0.8032**
aUAminpA	**0.3751**	0.3771	0.3773	0.3762	**0.5008**	0.5041	0.5022	**0.5008**	**0.2883**	0.2910	0.3057	0.2884	0.3919	0.3836	**0.3626**	0.3800
oes97	0.5164	0.5135	**0.5133**	0.5138	0.6118	0.6123	**0.6110**	0.6111	0.5779	0.5886	**0.5673**	0.5773	0.6971	0.6956	0.6984	**0.6910**
58028	0.2737	**0.2663**	0.2685	0.2690	0.4175	**0.4157**	0.4159	0.4158	0.3490	0.3542	**0.3469**	0.3490	**0.5129**	**0.5129**	**0.5129**	**0.5129**
15014	0.3877	**0.3841**	0.3856	0.3863	0.4837	0.4833	**0.4829**	0.4845	0.3900	0.4466	**0.3855**	0.3900	**0.5030**	**0.5030**	**0.5030**	**0.5030**
32511	0.7751	0.7722	**0.7686**	0.7725	0.8506	0.8549	**0.8504**	0.8506	0.8407	0.8481	**0.8228**	0.8427	0.9509	**0.9078**	0.9509	0.9100
15017	0.3306	0.3283	0.3291	**0.3263**	0.4112	0.4077	0.4093	**0.4076**	**0.3795**	0.3826	0.3796	**0.3795**	**0.5178**	0.5180	**0.5178**	0.5180
98502	0.6827	0.6816	**0.6814**	0.6817	0.8021	**0.8008**	0.8011	0.8022	0.7113	0.7262	**0.7015**	0.7113	0.9050	0.9106	0.9339	**0.8022**
92965	0.6193	0.6216	0.6169	**0.6156**	**0.6873**	0.6885	0.6885	0.6890	0.6867	0.6765	**0.6683**	0.6867	0.7452	**0.7287**	0.7452	0.7343
32314	0.5442	**0.5317**	0.5379	0.5362	0.5993	0.5988	**0.5974**	0.5990	0.5742	0.6107	**0.5587**	0.5742	0.7058	0.7345	**0.7044**	0.7317
13008	0.3072	**0.3056**	0.3070	0.3076	0.4680	**0.4642**	0.4663	**0.4642**	0.3782	0.3840	**0.3713**	0.3768	**0.6867**	**0.6867**	**0.6867**	**0.6867**
21114	0.2619	**0.2528**	0.2573	0.2593	0.4160	**0.4123**	0.4131	**0.4123**	0.3193	**0.3181**	0.3220	0.3193	**0.4630**	**0.4630**	**0.4630**	**0.4630**
85110	0.5896	0.5862	**0.5861**	0.5870	**0.6456**	0.6504	0.6476	**0.6456**	0.6580	0.7018	**0.6488**	0.6580	**0.6231**	0.6291	**0.6231**	0.6291

Base Target	RF				SVM				XGBoost				CART			
	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS
27311	0.6335	0.6312	**0.6281**	0.6324	**0.7113**	0.7132	0.7118	**0.7113**	**0.7308**	0.7375	0.7328	0.7321	0.8784	**0.8704**	0.8753	0.8971
98902	0.4620	0.4625	0.4624	**0.4590**	0.5414	0.5421	**0.5411**	0.5414	0.6167	**0.5985**	0.6076	0.6109	0.6249	**0.6246**	0.6249	**0.6246**
65032	0.5475	0.5465	0.5456	**0.5441**	0.6295	0.6305	**0.6291**	0.6295	0.5649	0.5830	**0.5551**	0.5663	0.6875	**0.6848**	0.6875	0.7035
92998	0.7167	0.7214	**0.7145**	0.7240	0.7643	0.7665	**0.7627**	0.7643	0.8360	0.8368	**0.8053**	0.8336	0.9486	0.9501	**0.9456**	0.9501
27108	0.5782	0.5733	0.5750	**0.5724**	0.6837	0.6860	**0.6833**	0.6836	0.5946	0.5991	**0.5721**	0.5911	0.6995	0.7032	0.6995	**0.6880**
53905	0.5535	0.5510	0.5489	**0.5468**	0.6770	0.6824	**0.6754**	0.6770	0.6166	0.6134	**0.5987**	0.6155	**0.7014**	**0.7014**	**0.7014**	**0.7014**
oes10	0.4070	0.4081	0.4070	**0.4057**	0.5464	0.5456	**0.5451**	0.5456	0.4691	0.4754	**0.4611**	0.4691	0.5976	0.6026	**0.5971**	0.5976
513021	0.3843	**0.3816**	0.3871	0.3836	0.5493	0.5495	**0.5486**	0.5497	0.4281	0.4487	**0.4203**	0.4281	**0.7180**	0.7420	0.7185	**0.7180**
292071	**0.3894**	0.3947	0.3894	0.3909	0.5443	**0.5409**	0.5415	0.5413	0.4445	0.4390	**0.4268**	0.4445	0.6441	**0.6426**	0.6441	0.6441
392021	0.3742	0.3737	0.3735	**0.3720**	0.5582	0.5590	**0.5578**	0.5590	0.4291	0.4393	**0.4216**	0.4291	**0.5601**	0.6089	**0.5601**	**0.5601**
151131	0.4804	0.4816	0.4788	**0.4782**	0.6073	0.6083	**0.6073**	0.6073	0.5790	0.5942	**0.5768**	0.5790	**0.6112**	0.6182	**0.6112**	**0.6112**
151141	0.3804	**0.3780**	0.3805	0.3816	0.6094	0.6090	**0.6080**	0.6093	0.3908	0.4042	**0.3767**	0.3908	0.5888	**0.5884**	0.5888	0.5888
291069	0.6260	0.6145	0.6177	**0.6130**	0.6439	0.6466	**0.6427**	0.6439	0.6521	0.6493	**0.6335**	0.6521	0.8138	0.8420	0.8138	**0.8138**
119032	0.3100	0.3169	0.3115	**0.3099**	0.5560	**0.5535**	0.5540	0.5537	0.4006	0.4215	**0.3950**	0.4006	**0.5006**	**0.5006**	**0.5006**	**0.5006**
432011	0.3514	**0.3482**	0.3519	0.3498	0.5072	0.5053	0.5055	**0.5053**	0.3799	0.3856	**0.3656**	0.3799	0.5308	**0.5196**	0.5224	0.5308
419022	0.5999	0.6065	0.5998	**0.5957**	0.6676	0.6675	**0.6657**	0.6676	0.7931	**0.7308**	0.7860	0.7931	**0.7764**	0.7826	**0.7764**	**0.7764**
292037	0.3548	0.3587	**0.3544**	0.3557	0.4938	**0.4914**	0.4918	0.4914	0.3840	**0.3666**	0.3775	0.3840	**0.4705**	**0.4705**	**0.4705**	**0.4705**
519061	0.4298	0.4368	0.4319	**0.4289**	0.5470	0.5472	**0.5462**	0.5479	0.4318	0.4710	**0.4279**	0.4318	**0.5696**	0.5699	**0.5696**	**0.5696**
291051	0.2254	0.2273	0.2253	**0.2251**	0.4016	**0.3989**	0.4004	0.3989	0.3053	**0.2978**	0.3013	0.3053	**0.3978**	**0.3978**	**0.3978**	**0.3978**
172141	0.4856	0.4908	0.4867	**0.4825**	0.5902	0.5901	**0.5889**	0.5903	**0.5795**	0.6369	0.5817	**0.5795**	**0.6334**	**0.6334**	**0.6334**	**0.6334**
431011	0.2344	0.2294	**0.2280**	0.2329	0.4494	**0.4475**	0.4481	**0.4475**	0.2007	0.2094	**0.1961**	0.2007	**0.4961**	**0.4961**	**0.4961**	**0.4961**
291127	0.4831	**0.4802**	0.4842	0.4861	0.5674	0.5670	**0.5666**	0.5676	0.5945	0.5917	**0.5832**	0.5945	**0.6494**	**0.6494**	**0.6494**	**0.6494**
412021	**0.4033**	0.4111	0.4106	0.4061	0.4489	**0.4483**	0.4485	0.4487	0.5122	0.5199	**0.5069**	0.5122	0.6010	**0.5801**	0.6010	0.6010
rf1	0.0782	0.0582	0.0731	**0.0534**	0.1215	0.1070	0.1151	**0.0997**	0.1274	0.1340	**0.1150**	0.1245	0.3523	0.3510	**0.3505**	0.3510
CHSI2	0.0150	0.0127	0.0135	**0.0117**	0.0781	0.0740	**0.0740**	0.0742	0.0325	0.0315	**0.0302**	0.0318	0.2834	0.2809	**0.2765**	0.2798
NASI2	0.4841	0.3399	0.4495	**0.3040**	0.2852	0.2370	0.2677	**0.2272**	0.7591	0.8178	**0.7022**	0.7426	0.7333	0.7193	0.7333	**0.7193**
EADM7	0.0182	0.0141	0.0157	**0.0136**	0.0909	0.0847	0.0857	**0.0825**	0.0416	0.0401	**0.0318**	0.0400	**0.3204**	0.3255	0.3215	0.3263

Base Target	RF				SVM				XGBoost				CART			
	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS
SCLM7	0.0190	0.0153	0.0180	**0.0146**	0.1048	0.0914	0.0988	**0.0884**	0.0436	0.0432	**0.0363**	0.0421	0.3073	0.3093	**0.3027**	0.3093
CLKM7	0.0225	0.0209	0.0220	**0.0206**	0.0910	0.0846	0.0885	**0.0803**	0.0379	0.0362	**0.0307**	0.0368	**0.3045**	**0.3045**	**0.3045**	0.3058
VALI2	0.0261	0.0237	0.0254	**0.0233**	0.1526	0.1210	0.1381	**0.0861**	0.0470	0.0451	**0.0368**	0.0452	0.3375	**0.3329**	0.3334	0.3330
NAPM7	0.0242	**0.0237**	0.0241	0.0247	0.0746	0.0741	0.0743	**0.0738**	0.0304	0.0317	**0.0283**	0.0317	**0.2309**	0.2340	0.2310	0.2335
DLDI4	0.0167	0.0151	0.0163	**0.0147**	0.0948	0.0891	0.0933	**0.0850**	0.0275	0.0262	**0.0240**	0.0261	**0.3014**	**0.3014**	**0.3014**	**0.3014**
rf2	0.0847	0.0784	0.0852	**0.0753**	0.1095	0.1064	0.1084	**0.1063**	0.1293	0.1320	**0.1212**	0.1282	0.3594	0.3720	**0.3552**	0.3568
CHSI2	0.0180	0.0157	0.0166	**0.0154**	0.0812	0.0797	**0.0796**	0.0801	0.0298	0.0288	**0.0262**	0.0292	0.2834	0.2851	**0.2725**	0.2817
NASI2	0.5151	0.4774	0.5252	**0.4544**	0.2491	**0.2456**	0.2485	0.2457	0.7876	0.8168	**0.7638**	0.7861	**0.7949**	0.9163	**0.7949**	**0.7949**
EADM7	0.0207	0.0179	0.0191	**0.0177**	0.0839	0.0824	0.0824	**0.0823**	0.0377	0.0363	**0.0266**	0.0366	0.3228	0.3159	**0.3096**	0.3155
SCLM7	0.0224	0.0200	0.0216	**0.0197**	0.0893	**0.0877**	0.0884	0.0880	0.0375	0.0366	**0.0339**	0.0366	0.2998	**0.2856**	0.2942	0.2919
CLKM7	0.0249	0.0239	0.0245	**0.0238**	0.0842	**0.0827**	0.0830	0.0828	0.0380	0.0363	**0.0303**	0.0361	**0.3045**	**0.3045**	**0.3045**	**0.3045**
VALI2	0.0288	0.0269	0.0279	**0.0267**	0.0959	0.0888	0.0944	**0.0879**	0.0454	0.0432	**0.0356**	0.0433	0.3375	**0.3329**	0.3334	0.3336
NAPM7	0.0277	0.0267	0.0273	**0.0265**	0.0934	**0.0924**	0.0929	0.0925	0.0311	0.0317	**0.0283**	0.0311	**0.2309**	0.2340	0.2310	**0.2309**
DLDI4	0.0202	0.0187	0.0194	**0.0183**	0.0990	0.0923	0.0983	**0.0915**	0.0274	0.0262	**0.0246**	0.0266	**0.3014**	**0.3014**	**0.3014**	**0.3014**
scm1d	0.2871	0.2758	0.2823	**0.2716**	0.3309	0.3232	0.3258	**0.3210**	0.3059	0.3074	**0.2802**	0.3056	0.5104	0.5029	**0.4867**	0.4985
LBL	0.2483	0.2415	0.2462	**0.2387**	0.3054	0.2994	0.3011	**0.2975**	0.2659	0.2665	**0.2443**	0.2659	0.4320	0.4301	**0.4216**	0.4309
MTLp2	0.2691	0.2624	0.2664	**0.2598**	0.3041	0.2981	0.3002	**0.2965**	0.2899	0.2924	**0.2696**	0.2899	0.4485	0.4441	**0.4383**	0.4419
MTLp3	0.2748	0.2646	0.2690	**0.2619**	0.3135	0.3072	0.3085	**0.3055**	0.3006	0.3014	**0.2710**	0.3006	0.5260	0.5156	**0.4828**	0.5130
MTLp4	0.2840	0.2730	0.2797	**0.2686**	0.3186	0.3118	0.3151	**0.3106**	0.2986	0.3005	**0.2742**	0.2986	0.5331	0.5324	**0.4971**	0.5238
MTLp5	0.2901	0.2702	0.2813	**0.2635**	0.3441	0.3307	0.3352	**0.3265**	0.3035	0.3022	**0.2796**	0.3022	0.6155	0.5984	**0.5573**	0.5908
MTLp6	0.3062	0.2861	0.2947	**0.2778**	0.3520	0.3410	0.3434	**0.3360**	0.3339	0.3330	**0.2934**	0.3330	0.6290	0.6052	**0.5540**	0.5897
MTLp7	0.2947	0.2747	0.2868	**0.2678**	0.3348	0.3202	0.3269	**0.3163**	0.3207	0.3197	**0.2859**	0.3197	0.6182	0.5892	**0.5553**	0.5701
MTLp8	0.3023	0.2820	0.2897	**0.2749**	0.3424	0.3275	0.3304	**0.3227**	0.3301	0.3304	**0.2909**	0.3296	0.6140	0.5945	**0.5520**	0.5821
MTLp9	0.2777	0.2676	0.2752	**0.2651**	0.3189	0.3127	0.3153	**0.3113**	0.2948	0.2964	**0.2732**	0.2948	0.4760	0.4743	0.4738	**0.4734**
MTLp10	0.2943	0.2852	0.2911	**0.2817**	0.3334	0.3262	0.3289	**0.3238**	0.3089	0.3117	**0.2896**	0.3089	0.4975	0.4992	**0.4939**	0.4970
MTLp11	0.2921	0.2835	0.2890	**0.2799**	0.3241	0.3185	0.3207	**0.3175**	0.3092	0.3119	**0.2837**	0.3092	**0.4699**	**0.4699**	**0.4699**	**0.4699**
MTLp12	0.3096	0.3018	0.3063	**0.2988**	0.3417	0.3352	0.3368	**0.3335**	0.3292	0.3323	**0.3052**	0.3292	**0.4817**	**0.4817**	**0.4817**	**0.4817**

Base Target	RF				SVM				XGBoost				CART			
	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS
MTLp13	0.2805	0.2734	0.2786	**0.2708**	0.3466	0.3416	0.3436	**0.3407**	0.2983	0.2998	**0.2767**	0.2983	0.4612	0.4577	**0.4560**	0.4577
MTLp14	0.3083	0.3002	0.3068	**0.2970**	0.3640	0.3590	0.3611	**0.3565**	0.3211	0.3245	**0.2992**	0.3211	0.4751	0.4735	**0.4723**	0.4735
MTLp15	0.2785	0.2707	0.2757	**0.2666**	0.3184	0.3137	0.3152	**0.3127**	0.2926	0.2962	**0.2697**	0.2926	0.4470	0.4402	0.4418	**0.4392**
MTLp16	0.2829	0.2758	0.2808	**0.2722**	0.3330	0.3285	0.3304	**0.3277**	0.2964	0.2990	**0.2775**	0.2964	0.4418	0.4413	**0.4390**	0.4411
scm20d	0.3648	0.3313	0.3347	**0.3116**	0.3972	0.3522	0.3474	**0.3301**	0.3701	0.3625	**0.3262**	0.3625	0.7307	0.7107	**0.6772**	0.7010
LBL	0.3253	0.2991	0.3051	**0.2874**	0.3403	0.3062	0.3088	**0.2951**	0.3318	0.3234	**0.2927**	0.3234	0.6823	0.6383	**0.6176**	0.6318
MTLp2A	0.3334	0.3047	0.3026	**0.2983**	0.3501	0.3166	0.3127	**0.3077**	0.3397	0.3326	**0.2995**	0.3326	0.6715	0.6539	**0.6181**	0.6509
MTLp3A	0.3414	0.3114	0.3129	**0.2962**	0.3634	0.3277	0.3225	**0.3101**	0.3509	0.3448	**0.3066**	0.3448	0.7036	0.6667	**0.6201**	0.6515
MTLp4A	0.3507	0.3217	0.3222	**0.3016**	0.3792	0.3411	0.3356	**0.3271**	0.3639	0.3562	**0.3170**	0.3562	0.7191	0.6927	**0.6506**	0.6743
MTLp5A	0.3748	0.3382	0.3436	**0.3138**	0.4272	0.3709	0.3640	**0.3357**	0.3718	0.3614	**0.3303**	0.3614	0.7630	0.7480	**0.7107**	0.7414
MTLp6A	0.3780	0.3383	0.3364	**0.3116**	0.4230	0.3670	0.3512	**0.3391**	0.3795	0.3683	**0.3330**	0.3683	0.7717	0.7386	**0.6870**	0.7266
MTLp7A	0.3728	0.3315	0.3377	**0.3027**	0.4233	0.3621	0.3631	**0.3274**	0.3763	0.3658	**0.3233**	0.3658	0.7374	0.7271	**0.6932**	0.7128
MTLp8A	0.3796	0.3371	0.3371	**0.3079**	0.4297	0.3617	0.3478	**0.3265**	0.3830	0.3753	**0.3326**	0.3753	0.7510	0.7178	**0.6919**	0.7111
MTLp9A	0.3580	0.3268	0.3301	**0.3140**	0.3895	0.3482	0.3432	**0.3289**	0.3592	0.3528	**0.3241**	0.3528	0.7121	0.7063	**0.6673**	0.6986
MTLp10A	0.3769	0.3457	0.3495	**0.3283**	0.4104	0.3657	0.3625	**0.3466**	0.3806	0.3726	**0.3383**	0.3726	0.7260	0.7233	**0.6807**	0.7195
MTLp11A	0.3724	0.3394	0.3445	**0.3200**	0.4019	0.3571	0.3585	**0.3334**	0.3749	0.3688	**0.3339**	0.3688	0.7419	0.7308	**0.7029**	0.7222
MTLp12A	0.3886	0.3565	0.3630	**0.3414**	0.4262	0.3809	0.3758	**0.3586**	0.3976	0.3930	**0.3519**	0.3930	0.7669	0.7483	**0.7124**	0.7334
MTLp13A	0.3680	0.3362	0.3434	**0.3185**	0.3973	0.3560	0.3512	**0.3355**	0.3725	0.3656	**0.3379**	0.3656	0.7209	0.7214	**0.6918**	0.7110
MTLp14A	0.3868	0.3533	0.3609	**0.3345**	0.4165	0.3796	0.3771	**0.3618**	0.3925	0.3853	**0.3477**	0.3853	0.7456	0.7279	**0.7032**	0.7171
MTLp15A	0.3584	0.3257	0.3336	**0.3028**	0.3874	0.3448	0.3450	**0.3191**	0.3641	0.3563	**0.3207**	0.3563	0.7346	0.7119	**0.7012**	0.7039
MTLp16A	0.3717	0.3350	0.3331	**0.3063**	0.3903	0.3497	0.3400	**0.3290**	0.3831	0.3773	**0.3296**	0.3773	0.7440	0.7179	**0.6860**	0.7098
edm	0.6721	0.6631	0.6661	**0.6478**	0.7699	0.7714	0.7667	**0.7646**	0.6793	0.7224	**0.6581**	0.6793	0.7042	0.7379	**0.6850**	0.7042
DFlow	0.6191	0.5579	0.6054	**0.5523**	**0.7003**	0.7256	0.7038	**0.7003**	0.6263	**0.5977**	0.6245	0.6263	0.7263	0.7764	**0.6955**	0.7263
DGap	**0.7250**	0.7682	0.7268	0.7434	0.8395	**0.8172**	0.8296	0.8289	0.7323	0.8471	**0.6918**	0.7323	0.6821	0.6994	**0.6745**	0.6821
sf1	1.0051	1.1280	1.0161	**1.0049**	0.9390	0.9477	**0.9250**	0.9390	1.3598	1.3479	**1.2478**	1.3374	**0.9667**	1.0278	0.9723	**0.9667**
c.class	1.0842	1.2316	1.1016	**1.0835**	1.0053	1.0019	**0.9932**	1.0053	1.4634	1.4283	**1.3097**	1.4294	**1.0535**	1.1160	1.0619	**1.0535**
m.class	**1.0988**	1.2367	1.1145	1.1011	1.0838	1.0980	**1.0531**	1.0838	1.5263	1.5279	**1.4081**	1.5273	**1.0719**	1.1467	1.0729	**1.0719**

Base Target	RF				SVM				XGBoost				CART			
	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS
x.class	0.8322	0.9156	0.8321	**0.8301**	**0.7279**	0.7434	0.7285	**0.7279**	1.0896	1.0876	**1.0255**	1.0554	**0.7747**	0.8207	0.7820	**0.7747**
sf2	0.8487	0.9410	0.8617	**0.8479**	0.7825	0.7787	0.7827	**0.7764**	1.0609	1.1126	**1.0109**	1.0609	0.8305	**0.8243**	0.8247	0.8302
c.class	**0.9881**	1.0944	1.0042	0.9890	1.0067	0.9924	1.0098	**0.9903**	1.2032	1.2130	**1.1685**	1.2032	0.9699	**0.9685**	0.9699	0.9706
m.class	**1.1009**	1.3221	1.1583	1.1021	0.9119	0.9150	**0.9103**	0.9147	1.5213	1.6333	**1.4281**	1.5213	1.0445	1.0425	**1.0233**	1.0428
x.class	0.4570	**0.4065**	0.4226	0.4525	0.4291	0.4287	0.4281	**0.4242**	0.4581	0.4914	**0.4361**	0.4581	0.4772	**0.4620**	0.4809	0.4774
jura	0.6061	0.5974	0.5969	**0.5921**	0.6409	0.6413	**0.6391**	0.6409	0.6383	0.6425	**0.6129**	0.6383	**0.7021**	0.7240	0.7042	**0.7021**
Cd	0.7056	**0.7011**	0.7018	0.7067	0.7103	**0.7052**	0.7059	0.7103	0.7516	0.7551	**0.7080**	0.7516	**0.8007**	0.8159	0.8020	**0.8007**
Co	0.5272	0.5289	0.5245	**0.5210**	**0.5918**	0.5965	0.5939	**0.5918**	**0.5400**	0.5406	0.5411	**0.5400**	**0.6059**	0.6243	0.6109	**0.6059**
Cu	0.5856	0.5623	0.5645	**0.5487**	0.6208	0.6221	**0.6175**	0.6208	0.6234	0.6319	**0.5897**	0.6234	**0.6997**	0.7317	**0.6997**	**0.6997**
wq	0.9066	0.9392	**0.9059**	0.9068	0.9630	**0.9535**	0.9581	0.9549	0.9828	1.0263	**0.9429**	0.9828	0.9786	1.0040	**0.9472**	0.9781
25400	0.9154	0.9638	0.9175	**0.9153**	0.9892	0.9835	**0.9830**	0.9868	0.9690	1.0199	**0.9402**	0.9690	0.9994	1.0046	**0.9489**	0.9994
29600	0.9972	1.0631	**0.9915**	0.9947	1.0549	**1.0487**	1.0526	1.0541	1.0734	1.1098	**1.0382**	1.0734	1.0693	1.1032	**1.0314**	1.0693
30400	0.9672	0.9908	**0.9590**	0.9661	0.9879	0.9852	**0.9842**	0.9900	1.0733	1.0995	**1.0266**	1.0733	1.0747	1.0982	**1.0124**	1.0747
33400	0.9018	0.9111	**0.8935**	0.9011	0.9394	0.9067	0.9345	**0.9046**	0.9775	1.0366	**0.9269**	0.9775	0.9275	0.9600	**0.9133**	0.9211
17300	0.8989	0.9270	0.9054	**0.8981**	0.9396	**0.9371**	0.9391	0.9379	0.9783	1.0416	**0.9269**	0.9783	0.9384	0.9723	**0.9253**	0.9384
19400	**0.8385**	0.8858	0.8394	0.8398	0.8879	0.9024	**0.8863**	0.8903	0.8862	0.9355	**0.8688**	0.8862	0.9064	0.9101	**0.8989**	0.9064
34500	0.9706	1.0108	**0.9630**	0.9694	1.0076	**0.9940**	1.0009	1.0033	1.1039	1.1396	**1.0300**	1.1039	1.0329	1.0769	**0.9838**	1.0329
38100	**0.9147**	0.9435	0.9185	0.9197	0.9773	0.9795	**0.9685**	0.9880	0.9931	1.0136	**0.9639**	0.9931	0.9930	1.0296	**0.9597**	0.9930
49700	**0.7856**	0.8067	0.7934	0.7901	0.8446	**0.8362**	0.8397	0.8396	0.8309	0.8725	**0.8162**	0.8309	0.8825	0.8993	**0.8346**	0.8825
50390	0.8814	0.9147	0.8836	**0.8791**	0.9811	0.9423	0.9747	**0.9376**	0.9414	0.9718	**0.8938**	0.9414	0.9750	0.9867	**0.9472**	0.9750
55800	0.9109	0.9466	**0.9078**	0.9120	0.9878	0.9838	**0.9801**	0.9906	1.0039	1.0577	**0.9575**	1.0039	0.9842	1.0267	**0.9680**	0.9842
57500	0.9166	0.9413	**0.9163**	0.9164	0.9627	**0.9473**	0.9557	0.9482	0.9939	1.0511	**0.9618**	0.9939	0.9987	1.0293	**0.9678**	0.9987
59300	0.9409	0.9668	**0.9374**	0.9397	1.0299	1.0057	1.0209	**1.0046**	1.0175	1.0625	**0.9765**	1.0175	1.0171	1.0456	**0.9702**	1.0171
37880	**0.8527**	0.8770	0.8569	0.8538	**0.8924**	0.8964	0.8929	0.8931	0.9170	0.9563	**0.8738**	0.9170	0.9006	0.9135	**0.8994**	0.9006
enb	0.1504	0.1173	0.1304	**0.1161**	0.2499	0.2173	0.2404	**0.1678**	0.0601	0.0617	0.0744	**0.0599**	**0.2906**	0.2971	0.2909	**0.2906**
Y1	0.1094	0.0519	0.0772	**0.0504**	0.2226	0.1866	0.2120	**0.1312**	**0.0337**	0.0371	0.0438	**0.0337**	**0.2577**	0.2663	**0.2577**	**0.2577**
Y2	0.1914	0.1828	0.1837	**0.1818**	0.2772	0.2479	0.2687	**0.2043**	0.0865	0.0863	0.1049	**0.0861**	**0.3236**	0.3278	0.3240	**0.3236**

Base Target	RF				SVM				XGBoost				CART			
	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS
slump	0.8365	0.8324	**0.8222**	0.8223	0.6924	0.6862	0.6819	**0.6784**	0.8518	0.8575	**0.7975**	0.8508	1.0254	1.0376	**1.0016**	1.0252
slump	1.0444	1.0475	**0.9899**	1.0444	0.9295	**0.8797**	0.8868	0.8829	1.1067	1.1228	**1.0225**	1.1052	1.2756	1.2665	1.2760	**1.2661**
flow	0.8806	0.9011	0.8809	**0.8737**	0.8718	0.8761	**0.8612**	0.8765	0.9946	0.9908	**0.9196**	0.9945	1.0594	1.1279	**1.0144**	1.0730
cpr_str	0.5846	0.5488	0.5959	**0.5487**	**0.2759**	0.3027	0.2976	**0.2759**	0.4542	0.4590	**0.4504**	0.4526	0.7412	0.7186	**0.7144**	0.7365
andro	0.7941	0.7349	0.7614	**0.6713**	1.1348	0.9243	1.0089	**0.7561**	0.4754	0.4705	**0.4130**	0.4676	0.6677	0.6343	**0.6118**	0.6284
Target	0.4058	0.3783	0.3897	**0.3615**	0.3603	0.3176	0.3295	**0.2796**	0.4278	0.4292	**0.3428**	0.4157	**0.5675**	0.5870	**0.5675**	**0.5675**
Target_2	2.2973	2.1807	2.1899	**1.8607**	4.1015	3.1101	3.5032	**2.2338**	0.1973	0.1985	**0.1527**	0.1991	**0.3889**	**0.3889**	**0.3889**	**0.3889**
Target_3	0.4339	**0.3633**	0.4152	0.3728	0.5497	0.4384	0.4979	**0.4090**	0.4063	0.4421	**0.3697**	0.4239	0.6483	0.5908	**0.5546**	0.5908
Target_4	0.4338	0.3713	0.4078	**0.3562**	0.5340	0.4256	0.4721	**0.3942**	0.4380	0.4157	**0.3834**	0.4279	0.6411	0.6028	**0.5912**	0.6028
Target_5	0.5904	0.5592	0.5897	**0.5454**	0.6281	0.6276	0.6222	**0.6118**	0.6398	0.6338	**0.6084**	0.6550	0.8127	0.8591	**0.8109**	0.8431
Target_6	0.6037	0.5566	0.5761	**0.5314**	0.6354	0.6262	0.6284	**0.6083**	0.7430	0.7039	**0.6210**	0.6840	0.9474	0.7771	**0.7577**	0.7771
osales	0.7577	**0.7275**	0.7332	0.7284	1.1726	1.1685	1.1702	**1.1685**	0.8404	0.8695	**0.7679**	0.8404	0.9560	0.9694	**0.7920**	0.9407
M1	0.7019	0.6739	0.6778	**0.6539**	1.6880	1.6850	1.6864	**1.6845**	1.1390	0.9948	**0.8039**	1.1390	0.8636	0.9061	**0.7106**	0.8846
M2	0.7578	**0.7206**	0.7253	0.7240	1.6460	**1.6390**	1.6427	1.6392	0.9360	1.0312	**0.8711**	0.9360	1.1590	1.2173	**0.9215**	1.2669
M3	0.7814	0.7514	0.7537	**0.7399**	0.9752	**0.9717**	0.9734	0.9721	**0.7568**	0.7627	0.8060	**0.7568**	0.8797	0.9064	**0.8320**	0.9093
M4	0.7510	0.7055	0.7117	**0.6836**	1.1845	**1.1801**	1.1815	1.1803	**0.7995**	0.8169	0.8084	**0.7995**	1.0512	0.9838	**0.7755**	1.0375
M5	0.7949	**0.7422**	0.7599	0.7740	1.3686	1.3658	1.3671	**1.3656**	0.9316	0.9601	**0.6987**	0.9316	1.0825	0.8963	**0.7389**	0.9239
M6	0.7299	**0.7046**	0.7132	0.7069	0.9761	**0.9721**	0.9740	0.9721	**0.6813**	0.7085	0.7311	**0.6813**	0.8886	0.8806	0.7618	**0.7551**
M7	0.7682	**0.7403**	0.7449	0.7523	0.9872	0.9838	0.9848	**0.9837**	0.7826	0.8379	**0.7412**	0.7826	0.9338	0.9618	**0.7908**	0.9027
M8	0.7542	**0.7374**	0.7405	0.7442	1.0623	1.0585	1.0601	**1.0583**	0.7935	0.8384	**0.7405**	0.7935	0.9168	1.0438	**0.8036**	0.9988
M9	0.7552	0.7256	0.7345	**0.7191**	1.1696	1.1662	1.1680	**1.1661**	0.8530	0.8994	**0.7681**	0.8530	0.9205	1.0910	**0.8117**	0.9671
M10	0.7570	**0.7339**	0.7411	0.7370	1.1674	**1.1611**	1.1634	1.1613	0.8080	0.8677	**0.7649**	0.8080	1.0246	0.9909	**0.7872**	0.9014
M11	0.7512	0.7328	**0.7300**	0.7391	0.8856	0.8821	0.8824	**0.8821**	0.7848	0.8402	**0.7357**	0.7848	0.8718	0.8704	**0.7761**	0.8645
M12	0.7894	**0.7617**	0.7660	0.7673	0.9605	0.9566	0.9579	**0.9566**	0.8184	0.8763	**0.7458**	0.8184	0.8804	0.8844	**0.7941**	0.8766
scfp	0.9263	0.9379	**0.8778**	0.8939	0.8242	0.8256	**0.8151**	0.8242	1.1163	1.1422	**0.9248**	1.1067	1.1075	1.0658	1.0882	**1.0214**
views	0.8228	**0.7799**	0.7817	0.7860	0.7596	**0.7377**	0.7462	0.7596	**0.8087**	0.8413	0.8877	**0.8087**	0.9850	0.9029	**0.8564**	0.9401
votes	0.7335	0.7501	**0.7214**	0.7289	0.7304	**0.7224**	0.7233	0.7304	0.8027	0.8128	**0.7632**	0.8027	0.8232	0.8108	**0.8076**	0.8410

Base / Target	RF				SVM				XGBoost				CART			
	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS	ST	MTRS	ERC	DSTARS
comments	1.2226	1.2836	**1.1304**	1.1668	0.9826	1.0167	**0.9758**	0.9826	1.7376	1.7726	**1.1234**	1.7087	1.5144	1.4836	1.6007	**1.2829**

TRABALHOS PUBLICADOS PELO AUTOR

Trabalhos publicados pelo autor durante o programa.

Publicações principais do trabalho.

1. Saulo Martiello Mastelini, Everton Jose Santana, Ricardo Cerri, Sylvio Barbon Jr., **DSTARS: A Multi-Target Deep Structure for Tracking Asynchronous Regressor Stack**, Brazilian Conference on Intelligent Systems – BRACIS, Outubro/2017, Sociedade Brasileira de Computação – SBC, págs. 19-24, ISBN 978-1-5386-2407-4, (Qualis CC 2017, B2).

2. Everton Jose Santana, Saulo Martiello Mastelini, Sylvio Barbon Jr., **Deep Regressor Stacking for Air Ticket Prices Prediction**, Simpósio Brasileiro de Sistemas de Informação – SBSI, Junho/2017, Sociedade Brasileira de Computação – SBC, págs. 25-31, ISBN 978-85-7669-76-5, (Qualis CC 2017, B2).

3. Felipe Kenji Nakano, Saulo Martiello Mastelini, Sylvio Barbon Jr., Ricardo Cerri, **Stacking Methods for Hierarchical Classification**, 16th IEEE International Conference On Machine Learning And Applications, IEEE, Dezembro/2017, págs. 289-296, ISBN 978-1-5386-1418-1, (Qualis CC 2017, B1).

4. Saulo Martiello Mastelini, Victor Guilherme Turrisi da Costa, Everton Jose Santana, Felipe Kenji Nakano, Rodrigo Capobianco Guido, Ricardo Cerri, Sylvio Barbon Jr, **Multi-output Tree Chaining: a interpretative modelling and lightweight multi-target approach**, Journal of Signal Processing Systems, Springer, R1 – *Minor Revisions*, (Qualis CC 2017, B1).

5. Felipe Kenji Nakano, Saulo Martiello Mastelini, Sylvio Barbon Jr., Ricardo Cerri, **Improving Hierarchical Classification of Transposable Elements using Deep Neural Networks**, International Joint Conference on Neural Networks – IJCNN, Julho/2018, IEEE, Aceito para publicação, (Qualis CC 2017, A1).

Publicações complementares.

1. Douglas F. Barbin, Saulo M. Mastelini, Sylvio Barbon Jr., Gabriel F.C. Campos, Ana Paula A.C. Barbon, Massami Shimokomaki, **Digital image analyses as an alternative tool for chicken quality assessment**, Biosystems Engineering, Abril/2016, Elsevier, págs. 85-93, (Qualis CC 2017, B3).

2. Saulo Martiello Mastelini, Matheus Camilo da Silva, Ana Paula Ayub da Costa, Sylvio Barbon Jr., **Marbling Grading Framework Applied on Meat Boutique Environment**, Simpósio Brasileiro de Sistemas de Informação – SBSI, Maio/2016, Sociedade Brasileira de Computação – SBC, págs. 542-549, ISBN 978-85-7669-319-2, (Qualis CC 2017, B2).

3. Gabriel Marques Tavares, Saulo Martiello Mastelini, Sylvio Barbon Jr., **User Classification on Online Social Networks by Post Frequency**, Simpósio Brasileiro de Sistemas de Informação – SBSI, Junho/2017, Sociedade Brasileira de Computação – SBC, págs. 464-471, ISBN 978-85-7669-376-5, (Qualis CC 2017, B2).

4. Ana Paula Ayub da Costa Barbon, Sylvio Barbon Jr., Gabriel Fillipe Centini Campos, José Luis Seixas Jr., Louise Manha Peres, Saulo Martiello, Mastelini, Nayara Andreo, Alessandro Ulrici, Ana Maria Bridi, **Development of a flexible Computer Vision System for marbling classification**, Computer and Electronics in Agriculture, Novembro/2017, Elsevier, págs. 536-544, (Qualis CC 2016, A2).

5. Ricardo Petri Silva, Gustavo Taiji Naozuka, Saulo Martiello Mastelini, Alan Salvany Felinto, **Automatic luminous reflections detector using Global Threshold with increased luminosity contrast in images**, Journal of Electronic Imaging, SPIE Digital Library, págs. 27-41, (Qualis CC 2016, A2).