UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

JOSÉ THIAGO HOLANDA DE ALCÂNTARA CABRAL

Um Framework para Seleção Dinâmica de Múltiplos Regressores Heterogêneos

Recife
2022

JOSÉ THIAGO HOLANDA DE ALCÂNTARA CABRAL

Um Framework para Seleção Dinâmica de Múltiplos Regressores Heterogêneos

Defesa de tese apresentada ao Programa de Pós-Graduação em Ciências da Computação da Universidade Federal de Pernambuco, como requisito para a obtenção do título de Doutor em Ciências da Computação.

Área de Concentração: Inteligência Computacional.

Orientador: Prof. Dr. Adriano Lorena Inácio de Oliveira

Recife
2022

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

C117f Cabral, José Thiago Holanda de Alcântara
Um framework para seleção dinâmica de múltiplos regressores heterogêneos / José Thiago Holanda de Alcântara Cabral. – 2022.
181 f.: fig., tab.

Orientador: Adriano Lorena Inácio de Oliveira.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2022.
Inclui referências e apêndices.

1. Inteligência computacional. 2. Aprendizagem de máquina. I. Oliveira, Adriano Lorena Inácio de (orientador). II. Título.

006.31 CDD (23. ed.) UFPE - CCEN 2023-36

José Thiago Holanda de Alcântara Cabral

“Um Framework para Seleção Dinâmica de Múltiplos Regressores Heterogêneos”

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Inteligência Computacional

Aprovado em: 17/11/2022.

_____

Orientador: Prof. Dr. Adriano Lorena Inacio de Oliveira

BANCA EXAMINADORA

_____

Prof. Dr. Fabio Queda Bueno da Silva
Centro de Informática / UFPE

_____

Profa. Dra. Renata Maria Cardoso Rodrigues de Souza
Centro de Informática / UFPE

_____

Prof. Dr. George Gomes Cabral
Centro de Computação / UFRPE

_____

Profa. Dra. Roberta Andrade de Araújo Fagundes
Escola Politécnica de Pernambuco / UPE

_____

Prof. Dr. Telmo de Menezes e Silva Filho
University of Bristol, Inglaterra

AGRADECIMENTOS

Agradeço, em primeiro lugar, a Deus, que fez com que meus objetivos fossem alcançados, durante todos os meus anos de estudos, por ter permitido que eu tivesse saúde e determinação para não desanimar durante a realização deste trabalho, pela minha vida, permitindo-me ultrapassar todos os obstáculos encontrados ao longo da realização deste trabalho.

A todos que estiveram comigo desde o início, no ano de 2013, quando iniciei o primeiro doutorado, o qual não consegui finalizar, somente agora tendo sido concluído. A todas as pessoas que foram muito importantes neste percurso, desde as que cursaram disciplinas comigo, até as mais próximas da minha vida pessoal. Em especial, a minha família, ao meu pai, José Augusto de Freitas Cabral, a minha mãe, Mabel Holanda de Alcântara Cabral, aos meus irmãos, Luciana Holanda de Alcântara Cabral e José Augusto de Freitas Cabral Júnior e, claro, aos meus quatro sobrinhos, Lucas, Mariana, Gabriel e Matheus. Além desses, meu cunhado, minha cunhada, tios e primos etc.

Ao professor Adriano Lorena Inácio de Oliveira, por ter sido meu orientador e ter desempenhado tal função com dedicação e amizade. Aos professores, pelas correções e ensinamentos que me permitiram apresentar um melhor desempenho no meu processo de formação profissional ao longo do curso, por todos os conselhos, pela ajuda e pela paciência com a qual guiaram o meu aprendizado. Em especial, à professora Renata Maria Cardoso Rodrigues de Souza, que foi tão presente durante toda a pesquisa, e ao professor Fábio Queda Bueno da Silva por ter aceitado trabalhar no desenvolvimento de um artigo importante e ter sido o presidente da banca de defesa desta tese.

Aos amigos próximos, mais antigos e recentes, que torceram tanto por mim, e que, em alguns casos, até oraram para que tudo desse certo desta vez. A todos aqueles que contribuíram, de alguma forma, para a realização deste trabalho. Em especial, aos colegas professores do Instituto Federal da Paraíba, Margareth, Keiteane Souza, Marcelo Garcia, Thiago José, Alexsandro Cunha, Juliana Dantas e Sérgio, professor de português, além desses, a minha psicóloga, Edvânia Galindo, que me acompanhou durante todo o processo.

A todos que participaram, direta ou indiretamente do desenvolvimento deste trabalho de pesquisa, enriquecendo o meu processo de aprendizado. Às pessoas com quem convivi ao longo desses anos de curso, que me incentivaram e que certamente tiveram impacto na minha formação acadêmica.

RESUMO

O uso de Aprendizagem de Máquina (AM) tem sido cada vez mais comum em diversas áreas do conhecimento. A engenharia de *software*, assim como os estudos de desempenhos educacionais, são exemplos de áreas que têm aderido às técnicas de AM. Os métodos usados vão desde os individuais, que geram saídas a partir de um modelo, até os Sistema de Múltiplos Modelos (SMM). As estratégias usadas nessas combinações têm sido um importante tópico de pesquisa em AM, visto que o uso de múltiplos modelos tem diminuído o erro e a variância das previsões em relação aos modelos solos. Estas estratégias variam entre diferentes formas de selecionar e integrar os modelos selecionados. Partindo deste princípio, este trabalho propõe um *framework* para seleção dinâmica de múltiplos modelos de regressão heterogêneos. A partir de um conjunto de dados, um subconjunto de exemplos é separado para formar os dados de treinamento, e o restante dos dados constituem a base de teste. Diversos algoritmos de regressão são treinados e validados na base de treinamento. Três diferentes modelos são selecionados a partir do desempenho individual de cada algoritmo e, a este grupo de modelos daremos o nome de Conjunto Básico (CB). Em seguida, é criada uma base de treinamento para problemas de classificação. Base esta que consiste em identificar, para cada exemplo de treinamento, o modelo de regressão pertencente ao CB de melhor desempenho. A partir deste conjunto de dados, diferentes classificadores são gerados e avaliados. Consequentemente, um conjunto de modelos de classificação é definido e nomeado de Conjunto de Seletores (CS), de forma que, ao fim da fase de treinamento e validação, na fase de teste, os melhores classificadores (CS) são usados para selecionar de forma dinâmica e ponderada os modelos de regressão do CB. A previsão final é dada pela combinação das saídas destes regressores selecionados dinamicamente pelos classificadores. Posteriormente, a fim de validar a proposta, análises experimentais em duas áreas do conhecimento são apresentadas. Foram investigados dois repositórios de Estimativa de Esforço de *Software* (EES) e seis bases de dados educacionais. Além de verificar o desempenho dos métodos gerados a partir do *framework* proposto, também foi analisado o comportamento destes métodos quanto aos critérios de seleção dinâmica, e a possibilidade da existência de correlações entre as metas-características das bases de dados com o desempenho do *framework*. Os experimentos utilizaram seis métricas para análise dos resultados, sendo a média do erro absoluto usada para fins de testes estatísticos. Os resultados demonstraram que os métodos gerados por intermédio do *framework* proposto superaram, na maioria das vezes, os modelos individuais, assim como diferentes estratégias de combinações desses modelos.

Palavras-chaves: aprendizagem de máquina; sistemas de múltiplos modelos heterogêneos; seleção dinâmica; seleção dinâmica de múltiplos modelos; modelos de regressão.

ABSTRACT

The use of Machine Learning (ML) has been increasingly common in several areas of knowledge. Software engineering, as well as educational performance studies, are examples of areas that have adhered to ML techniques. The methods used range from the individual ones, which generate outputs from a model, to the Multiple Model System. The strategies used in these combinations have been an important research topic in AM, since the use of multiple models has reduced the error and variance of predictions in relation to solo models. These strategies vary between different ways of selecting and integrating the selected models. Based on this principle, this work proposes a framework for dynamic selection of multiple heterogeneous regression models. From a data set, a subset of examples is separated to form the training data, and the rest of the data constitute the test base. Several regression algorithms are trained and validated on the training base. Three different models are selected based on the individual performance of each algorithm, and this group of models will be named Basic Set (CB). Then, a training base for classification problems is created. This basis consists of identifying, for each training example, the regression model belonging to the best performing CB. From this dataset, different classifiers are generated and evaluated. Consequently, a set of classification models is defined and named Set of Selectors (CS), so that, at the end of the training and validation phase, in the testing phase, the best classifiers (CS) are used to select in a way dynamic and weighted CB regression models. The final prediction is given by combining the outputs of these variable dynamically selected by the classifiers. Subsequently, in order to validate the proposal, experimental analyzes in two areas of knowledge are presented. Two Software Effort Estimation (SEE) repositories and six educational databases were investigated. In addition to verifying the performance of the methods generated from the proposed framework, the behavior of these methods was also analyzed regarding the dynamic selection criteria, and the possibility of correlations between the meta-characteristics of the databases and the performance of the framework. The experiments used six metrics to analyze the results, with the mean absolute error being used for statistical test purposes. The results showed that the methods generated through the proposed framework outperformed, in most cases, the individual models, as well as different strategies for combining these models.

Keywords: machine learning. heterogeneous multi-model systems; dynamic selection; dynamic ensemble selection; regression models.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

τ_c	Base de Treinamento dos Classificadores
τ_r	Base de Treinamento dos Regressores
τ_t	Base de Teste
τ_v	Base de Validação
AASC-NUCCI	*Author of the Adaptive Selection of Classifiers in Bug Predicting*
AE	*Absolute Error*
AM	Aprendizagem de Máquina
ANN	*Artificial Neural Network*
ANOVA	Análise de Variância
ANOVA-MR	Análise de Variância para Medidas Repetidas
BEM	*Basic Ensemble Method*
BN	*Bayesian Network*
CB	Conjunto Básico
CBR	*Case-Based Reasoning*
CS	Conjunto de Seletores
DCS	*Dynamic Classifier Selection*
DCS-LA	*Dynamic Classifier Selection By Local Accuracy*
DCS-LAW	*Dynamic Classifier Selection By Local Accuracy Weighted*
DES	*Dynamic Ensemble Selection*
DS	*Dynamic Selection*
DT	*Decision Trees*
DW	*Dynamic Weghting*
DWS	*Dynamic Weighting Selection*
EDM	*Education Data Mining*
EEE	*Ensemble Effort Estimation*
EES	Estimativa de Esforço de *Software*
EL	*Ensemble Learning*
ES	Engenharia de *Software*
GEM	Generalized Ensemble Method
GP	*Gaussian Process*

IA	Inteligência Artificial
INEP	Instituto Nacional de Educação e Pesquisa
IRWM	*Inversed Ranking Weighted Mean*
ISBSG	*International Software Benchmarking Standard Group*
KNN	*K-Nearest Neighbor*
KNORA	*K-Nearest Oracle*
KNORA-UNION-W	*K-Nearest Oracle Union - Weighted*
KNORA-ELIMINATE-W	*K-Nearest Oracle Eliminate - Weighted*
KNORA-U	*K-Nearest Oracle Union*
KNORA-E	*K-Nearest Oracle Eliminate*
LCA	*Local Class Accuracy*
LMS	*Least Median Squared*
LSD	*Least Significant Difference*
M5P	*M5 Base*
M5R	*M5 Rules*
MAE	*Mean Absolute Error*
MdMRE	*Median Magnitude Relative Error*
MedAE	*Median Absolute Error*
MLP	*Multi Layer Perceptron*
MLR	*Multi Linear Regression*
MMRE	*Mean Magnitude Relative Error*
MQE	*Mean Quadratic Error*
MRE	*Magnitude Relative Error*
MRE-*ADJUSTED*	*Magnitude Relative Error - Adjusted*
OLA	*Overall Local Accuracy*
PEETACO-DES	*Process of Evaluating Estimates Through Algorithm Combinations - Dynamic Ensemble Selection*
PEETACO-DS	*Process of Evaluating Estimates Through Algorithm Combinations - Dynamic Selection*
PRED	*Percentage Relative Error Deviation*

PROMISE	*Predictor Models In Software Engineering Repository*
RBFNN	*Radial Basis Function Neural Network*
RNA	Redes Neurais Artificiais
RT	*Regression Trees*
SA	*Standard Accuracy*
SES	*Static Ensemble Selection*
SMM	Sistema de Múltiplos Modelos
SVM	*Support Vector Machine*
SVR	*Support Vector Regression*

SUMÁRIO

1 INTRODUÇÃO

A estimativa de esforço é um desafio na área de gerenciamento de projetos de *software*. O processo de estimativa é definido como um processo de previsão do esforço requerido para desenvolver um sistema de *software* (WEN et al., 2012). Neste sentido, o maior desafio deste processo é alcançar estimativas de esforço que não contribuam para o cancelamento do projeto. Algumas razões que levam os projetos de *software* a falharem são: (i) metas de projeto irrealísticas ou não articuladas, (ii) estimativas imprecisas quanto aos recursos necessários e, (iii) incapacidade de lidar com a complexidade do projeto (CHARETTE, 2005).

De forma semelhante às estimativas em Estimativa de Esforço de *Software* (EES), a mineração de dados educacionais é outra área em que pesquisadores têm tentado estimar valores em busca de identificar padrões nas taxas de aprendizados dos estudantes. Neste sentido, o desenvolvimento e a aplicação de métodos automatizados para detectar padrões em dados educacionais têm sido investigado em diversos estudos (ROMERO; VENTURA, 2010a).

1.1 CONTEXTUALIZAÇÃO

Pesquisadores da área de EES têm proposto vários modelos para tentar melhorar a precisão das estimativas em projetos de *software*. Diferentes técnicas são conhecidas na literatura, entre elas: (i) técnicas baseadas na visão dos especialistas (TRENDOWICZ, 2014); (ii) técnicas paramétricas (BOEHM, 1981; AZZEH et al., 2018); (iii) técnicas baseadas em analogia (WALKERDEN; JEFFERY, 1999; ABNANE et al., 2019); (iv) modelos estatísticos (PHANNACHITTA; MATSUMOTO, 2019; CARVALHO et al., 2020) e (v) métodos baseados em Aprendizagem de Máquina (AM) (OLIVEIRA, 2006; AMARAL et al., 2019; RAO; RAO, 2020). Em um contexto de dados distinto, mas com objetivos semelhantes, a mineração de dados educacionais (*Education Data Mining* (EDM)) está imergindo como uma área de pesquisa que busca analisar dados, a fim de estudar e responder questões de pesquisa da área de educação (BEEMER et al., 2017a; HELLAS et al., 2018; NASCIMENTO; FAGUNDES; MACIEL, 2019; SANTOS; RODRíGUEZ; PINTO-LLORENTE, 2020). Por esta razão, o uso de modelos de AM tem atraído a atenção destas comunidades (IDRI; HOSNI; ABRAN, 2016a; ROMERO; VENTURA, 2010a).

A escolha de um modelo de AM que alcance um bom desempenho para vários problemas tem sido considerada uma tarefa difícil, já que os modelos individuais de AM comportam-se de maneiras diferentes para diferentes conjuntos de dados, o que leva a serem mais instáveis (KOCAGUNELI; MENZIES; KEUNG, 2012). De acordo com o teorema *No Free Lunch*, não existe um método que seja bom para todos os problemas (WOLPERT;

MACREADY, 1997). Deste modo, é mais proveitoso determinar o melhor método em um contexto particular do que induzir um único de forma geral (SHEPPERD; KADODA, 2001). Consequentemente, há um custo adicional para alcançar resultados ótimos com um único modelo. Este custo concentra-se na necessidade de encontrar o modelo mais estável. Uma forma de superar essa dificuldade e reduzir a instabilidade de desempenho destes modelos é utilizar Sistema de Múltiplos Modelos (SMM) (*Ensembles*) (KOCAGUNELI; MENZIES; KEUNG, 2012; IDRI; HOSNI; ABRAN, 2016a; DIETTERICH, 2000; ROMERO; VENTURA, 2013; SANTOS; RODRíGUEZ; PINTO-LLORENTE, 2020). Os SMM tendem a ser mais confiáveis e apresentam menos instabilidade nos resultados quando comparado aos modelos individuais (KOCAGUNELI; MENZIES; KEUNG, 2012). As vantagens deles em relação aos modelos individuais são relatadas quanto à maior robustez e melhor desempenho nos resultados (MOREIRA et al., 2012; CRUZ; SABOURIN; CAVALCANTI, 2018). A construção de um SMM envolve basicamente as fases de geração e integração. Há também a fase de corte, usada opcionalmente entre a geração e a integração.

Na fase de geração, os sistemas podem ser homogêneos ou heterogêneos e, de acordo com (DIETTERICH, 2000), um *ensemble* é homogêneo quando ele usa diversos modelos a partir de um único algoritmo, enquanto os *ensembles* são ditos heterogêneos quando utilizam modelos oriundos de dois ou mais algoritmos distintos para gerar a saída final. Observe que *ensembles* heterogêneos podem usar *ensembles* homogêneos como modelos individuais, uma vez que a qualidade dos ensembles está relacionada ao desempenho e diversidade dos modelos e, neste sentido, combinar bons modelos de diferentes algoritmos parece ser uma estratégia promissora. Existem basicamente duas abordagens usadas na fase de integração: (i) a fusão, e (ii) a seleção.

Na integração, as saídas de cada modelo são combinadas de forma linear (Ex. aritmética) ou não linear (ex. Redes neurais), enquanto a seleção pode ser estática ou dinâmica. Na seleção estática (*Static Ensemble Selection* (SES)), um subconjunto de modelos é utilizado para estimar a saída de todas as instâncias do conjunto de dados avaliado. Este subconjunto de modelos é selecionado na fase de treinamento, no entanto, as instâncias são associadas a diferentes níveis de dificuldade para a previsão. Neste sentido, a literatura tem mostrado que melhores resultados podem ser obtidos com a seleção de um subconjunto de modelos específico para cada instância avaliada (BRITTO; SABOURIN; OLIVEIRA, 2014). Este tipo de seleção é nomeada *Dynamic Ensemble Selection* (DES).

A seleção de modelos, muitas vezes, é abordada como um tipo de fusão, em que algumas saídas recebem peso 0 (zero), o que significa que o modelo não foi selecionado para a integração final. Neste trabalho, continuaremos utilizando o termo *ensemble* para se referir aos *ensembles* homogêneos ou aos heterogêneos. Para um contexto geral, será utilizado o termo SMM, no entanto, ambos têm o mesmo significado.

Revisões da literatura têm evidenciado o crescimento do uso de métodos de AM aplicados às áreas de EES e EDM. Os estudos (WEN et al., 2012; JORGENSEN; SHEPPERD, 2007;

USMAN et al., 2014) identificaram diversos modelos de AM aplicados em EES, enquanto (IDRI; HOSNI; ABRAN, 2016a; IDRI; HOSNI; ABRAN, 2016b) apresentou o crescimento do uso de SMM na área. Em paralelo, pesquisadores da área de EDM têm aplicado o uso de métodos de AM em dados educacionais (ROMERO; VENTURA, 2010a; ROMERO; VENTURA, 2013). Adicionalmente, (MOREIRA et al., 2012) apresentaram em um *survey* a utilização de *ensembles* de regressores, tendo sido destacado no trabalho as vantagens de adotar um SMM para problemas de regressão.

Quanto às estratégias de construção de *ensembles*, o uso de seleção dinâmica tem melhorado o desempenho final do SMM (WOODS; KEGELMEYER; BOWYER, 1997; CRUZ; SABOURIN; CAVALCANTI, 2018). Em (BRITTO; SABOURIN; OLIVEIRA, 2014) foi realizada uma revisão abrangente na literatura, quanto aos métodos de seleção dinâmica aplicados. A partir deste estudo, foi identificado que selecionar dinamicamente regressores heterogêneos ainda é uma área pouco investigada, visto que a ampla maioria dos métodos desenvolvidos abordam *ensembles* homogêneos e problemas de classificação. Porém tanto na seleção dinâmica simples quanto na seleção dinâmica de múltiplos modelos, percebe-se fortes tendências de crescimento da área, devido aos avanços alcançados em relação ao desempenho dos métodos, principalmente com o uso de DES (KO; SABOURIN; BRITTO JR., 2008). Alguns estudos apresentam as melhorias desta abordagem e mostram como ela tem alcançado bons resultados em diferentes experimentos (KO; SABOURIN; JR., 2007; CAVALIN; SABOURIN; SUEN, 2013; CRUZ; SABOURIN; CAVALCANTI, 2015; CRUZ; SABOURIN; CAVALCANTI, 2017; MOURA; CAVALCANTI; OLIVEIRA, 2021).

Diante do que foi abordado nesta seção, pode-se perceber que: (i) a área de *Ensemble Effort Estimation* (EEE) ainda tem problemas em aberto, como afirma (IDRI; HOSNI; ABRAN, 2016a), e que existem estudos relacionados à aplicação de métodos de AM na educação, como apresentado na revisão (ROMERO; VENTURA, 2010a); (ii) múltiplos modelos de regressão tendem a obter melhorias em relação aos modelos individuais (MOREIRA et al., 2012; IDRI; HOSNI; ABRAN, 2016b), assim como ocorre em problemas de classificação (CRUZ; SABOURIN; CAVALCANTI, 2017); (iii) métodos de seleção dinâmica apresentaram avanços em problemas de classificação (BRITTO; SABOURIN; OLIVEIRA, 2014); (iv) o uso de seleção dinâmica de múltiplos modelos já obteve performances melhores do que a seleção estática (KO; SABOURIN; BRITTO JR., 2008); e, finalmente (v), a abordagem para geração de *ensembles* heterogêneos é capaz de alcançar melhores resultados do que os alcançados pelos modelos individuais (IDRI; HOSNI; ABRAN, 2016a). Assim, considerando o potencial dos *ensembles* heterogêneos, que por vezes são melhores do que os homogêneos, e os avanços alcançados com a seleção dinâmica, é plausível a investigação do uso dessas duas abordagens. No entanto, apesar de existirem diversos métodos de seleção dinâmica na literatura (WOODS; KEGELMEYER; BOWYER, 1997; KO; SABOURIN; JR., 2007), a seleção dinâmica de regressores heterogêneos não tem sido citada nas revisões desta área (BRITTO; SABOURIN; OLIVEIRA, 2014; CAVALIN; SABOURIN; SUEN, 2013; MOREIRA et al.,

2012; KO; SABOURIN; BRITTO JR., 2008). Além disso, o *framework* proposto aborda o uso de diferentes critérios para selecionar um subconjunto de modelos dinâmicos, sendo estes critérios baseados no poder de aprendizado de diferentes classificadores.

Partindo desses princípios, até onde se tem conhecimento, a literatura não apresentou a utilização de múltiplos modelos de regressão heterogêneos selecionados dinamicamente através de classificadores com diferentes critérios. Portanto, considerando que a abordagem de *ensembles* é promissora nas áreas de EES e de desempenho educacional, e que DES tem alcançado resultados significativos em outros domínios, é abordado neste trabalho a união desses dois conceitos: (i) sistemas de múltiplos modelos de AM e (ii) seleção dinâmica. Em resumo, o *framework* proposto é descrito pelas etapas:

1. Seleção estática de múltiplos regressores heterogêneos (Conjunto Básico (CB));
2. Seleção estática de múltiplos classificadores heterogêneos (Conjunto de Seletores (CS));
3. Seleção dinâmica de regressores do CB através do CS;
4. Integração das saídas dos regressores selecionados dinamicamente;
5. Aplicação da média ponderada dinamicamente para saída final.

1.2 PROBLEMA DE PESQUISA

São abordadas nesse trabalho as seguintes questões de pesquisa:

1. Os diferentes tipos de *ensembles*, realmente, superam os métodos individuais em problemas de EES e EDM?

   Investiga se os SMM, de fato, superam os métodos bases.

2. Qual o desempenho do *framework* proposto e dos diferentes tipos de *ensembles* em problemas de EES e EDM?

   Compara os tipos de *ensemble* (homogêneo/heterogêneo) e o desempenho dos métodos gerados a partir do *framework* proposto em relação a estes *ensembles*.

3. Qual o desempenho do *framework* proposto e dos métodos de seleção dinâmica no contexto de EES e EDM?

   Analisa o uso de seleção dinâmica simples e de múltiplos modelos e se o *framework* proposto alcançou melhores resultados do que métodos conhecidos na literatura.

4. Existe algum critério de seleção dinâmica capaz de sobressair quanto à capacidade de selecionar regressores heterogêneos?

   Busca associar diferentes classificadores ao domínio dos dados analisado.

5. Existem meta-características nas bases de dados ou entre os modelos individuais que favoreçam o uso do *framework* proposto?

   Realiza uma análise associativa entre as metas-características das bases de dados (Ex. número de instâncias, quantidade de atributos etc.) e o desempenho do *framework* proposto.

As respostas a essas questões são apresentadas na Seção 6.4. As evidências se dão a partir dos resultados (Capítulo 6), que podem ser facilmente replicados.

1.3 OBJETIVOS

O objetivo geral deste trabalho é desenvolver um *framework* de seleção dinâmica de múltiplos modelos de regressão. Em seguida, avaliar instâncias do método proposto em problemas de EES e de EDM.

Os objetivos específicos buscam:

- Encontrar os modelos de AM que são mais acurados dentro do contexto de EES e EDM;
- Investigar os resultados alcançados por SMM em EES e EDM;
- Perquirir a utilização da seleção dinâmica de múltiplos modelos em EES e EDM;
- Pesquisar diferentes critérios de seleção dinâmica de modelos de regressão heterogêneos;
- Identificar os modelos de classificação mais adequados para selecionar regressores dinamicamente dentro da arquitetura do *framework* proposto em EES e EDM;
- Avaliar a correlação entre as meta-características dos dados e o desempenho do *framework* proposto expandindo as possibilidades de trabalhos futuros.

1.4 METODOLOGIA

Esta pesquisa aborda o uso de diversos algoritmos de AM, os quais foram aplicados a problemas de EES e de EDM. Para isso, foram realizadas buscas na literatura das duas áreas e diversos estudos encontrados apresentaram a aplicação de métodos individuais de AM e de SMM na previsão de esforço de *software* e nas taxas de desempenhos dos estudantes, podemos destacar (IDRI; HOSNI; ABRAN, 2016a; ROMERO; VENTURA, 2010a). Semelhante aos dados de EES, os educacionais também se referem a problemas de regressão. A partir destes estudos foram selecionados os algoritmos mais comuns nessas áreas. Além deles, alguns algoritmos disponíveis na biblioteca Weka 3.6.10 também foram

avaliados. O desempenho desses métodos foi analisado de maneira individual e combinada (estática e dinamicamente), sendo utilizados dois repositórios de EES, que somados totalizam 9 conjuntos de dados, e seis conjuntos de dados educacionais, todos avaliados individualmente.

O objetivo da análise de dados de problemas distintos é verificar se os resultados alcançados pelo *framework* proposto são semelhantes. Os dados de EES e educacionais investigados possuem caraterísticas diferentes. Por exemplo: (i) as variações dos valores das variáveis dependentes são menores nas bases educacionais; e (ii) os dados educacionais não contém dados ausentes, o que ocorreu em um dos repositórios de EES.

Neste sentido, três análises experimentais foram realizadas. A primeira utilizou 1466 projetos de *softwares* oriundos do repositório do *International Software Benchmarking Standard Group* (ISBSG) (ABRAN, 2015). A segunda análise usou 8 bases de dados de EES disponibilizadas no *Predictor Models In Software Engineering Repository* (PROMISE) (SHIRABAD; MENZIES, 2005), e finalmente, a terceira, abordou 6 conjuntos de dados educacionais que também foram avaliadas nos estudos (NASCIMENTO; FAGUNDES; MACIEL, 2019; BEEMER et al., 2017a). Durante cada experimento, os conjuntos de dados investigados foram divididos em bases de treinamento (treinamento/validação) e bases de testes.

A métrica de custo utilizada para avaliar a performance estatística dos modelos foi a média do erro absoluto ou a média do *ranking* de posições de cada método a partir do erro absoluto. Em cada experimento foi analisado se existiam diferenças significativas entre os métodos avaliados.

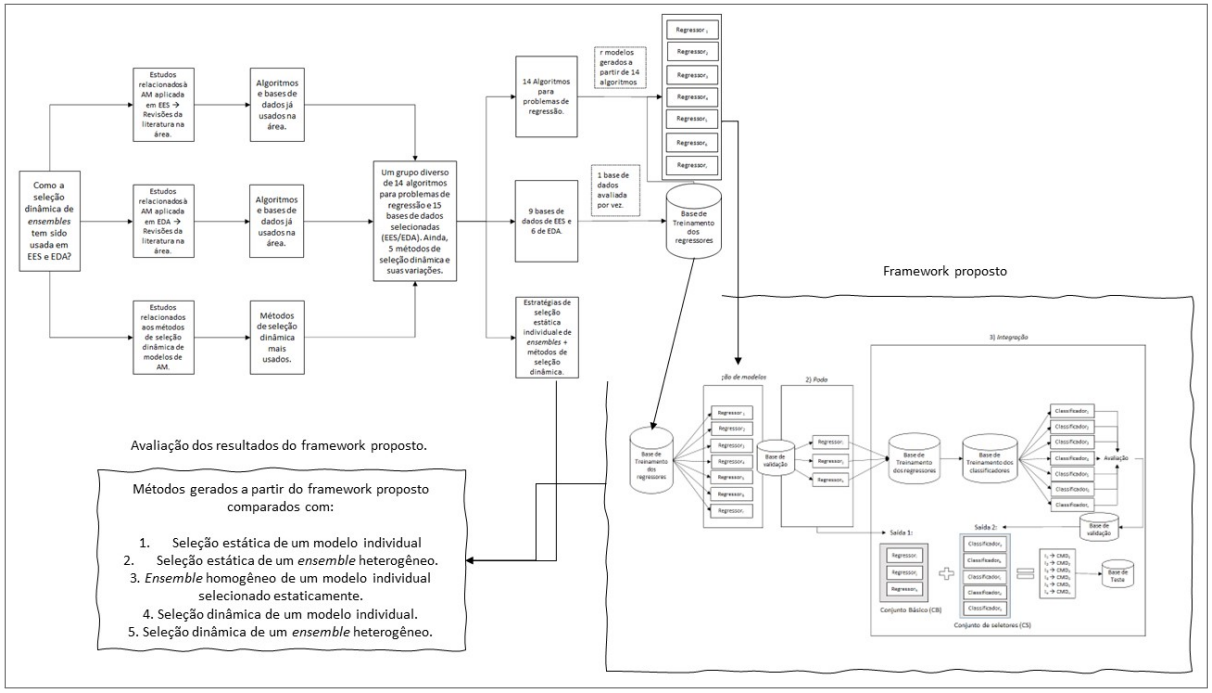
Na Figura 1 é ilustrada uma visão geral de como esta pesquisa foi realizada, desde o levantamento dos estudos, até os resultados apresentados. A princípio, revisões da literatura de seleção dinâmica permitiram encontrar os métodos mais comuns na área, enquanto as revisões de SMM aplicados à EES e a EDM deram suporte à identificação dos algoritmos de regressão mais utilizados para estimar projetos de *softwares* e taxas de desempenho dos alunos. Além dos algoritmos, foi preciso encontrar conjuntos de dados das duas áreas, que fossem comumente usados em estudos anteriores ou que caracterizassem o domínio do problema. Uma vez definido o grupo de algoritmos e o conjunto de dados a ser avaliado, uma parte deste conjunto foi separada para treinar e outra para testar os algoritmos.

A base de treinamento contém as instâncias usadas para o aprendizado e validação dos modelos. Um conjunto de r regressores heterogêneos são gerados a partir da Base de Treinamento dos Regressores (τ_r) e validados na Base de Validação (τ_v) e, os três melhores, são selecionados para formar o CB. Uma parte dos dados de treinamento (τ_r) é usada para comparar o desempenho de cada regressor do CB na própria τ_r. A partir do desempenho de cada modelo individual, é criada a Base de Treinamento dos Classificadores (τ_c), que contém as mesmas variáveis independentes da τ_r, mas difere quanto à variável dependente. Na τ_c, essa variável é categórica e identifica o melhor modelo de regressão para cada

instância. Em seguida, c classificadores são criados a partir dos dados da τ_c, sendo estes validados na τ_v, consequentemente, os melhores classificadores avaliados compõem o CS. Finalmente, os modelos do CB e do CS são usados para prever as saídas nas instâncias da Base de Teste (τ_t). Mais detalhes serão apresentados no Capítulo 4.

Os métodos gerados a partir do *framework* proposto foram comparados com: (i) a seleção estática individual; (ii) a seleção estática de *ensembles* heterogêneos; (iii) a seleção estática de *ensembles* homogêneos; (iv) a seleção dinâmica de um modelo *Dynamic Selection* (DS); e (v) a seleção dinâmica de múltiplos modelos (DES).

Figura 1 – Metodologia desenvolvida para avaliar o *framework* proposto



Fonte: O autor (2022)

1.5 PUBLICAÇÕES

Até a presente data, três artigos, sendo um em conferência internacional (*International Conference Tools Artificial Intelligence*) e dois em periódico internacional (*The Journal of Systems & Software*), foram publicados como resultado desta pesquisa.

1. Heterogeneous Ensemble Dynamic Selection for Software Development Effort Estimation (CABRAL et al., 2017)

2. Ensemble Effort Estimation Using Dynamic Selection (CABRAL; OLIVEIRA, 2021)

3. Ensemble Effort Estimation: an updated and extended systematic literature review (CABRAL; OLIVEIRA, 2021)

Neste sentido, como forma de contribuição deste trabalho, acreditamos que os seguintes fatores são as principais razões para se ter alcançado os resultados que serão apresentados no Capítulo 6:

- O desempenho avaliado antecipadamente em um conjunto de validação para selecionar os melhores modelos individuais, ou seja, o uso de seleção estática simples antes da seleção dinâmica;
- O desempenho avaliado antecipadamente para selecionar o melhor CB, ou seja, aplicação de seleção estática de um conjunto;
- Avaliação do desempenho de classificadores para construir o melhor CS para cada conjunto de dados, ou seja, seleção estática de classificadores para selecionar regressores dinamicamente;
- O uso de múltiplos modelos heterogêneos para diminuir a variância dos modelos individuais;
- O uso de seleção dinâmica para buscar o melhor padrão de estimativa;
- Adoção de diferentes critérios para selecionar dinamicamente o melhor *ensemble* para cada instância de teste.
- A possibilidade de gerar pesos dinâmicos para cada modelo de regressão selecionado.

1.6 ESTRUTURA

Os capítulos restantes desta tese encontram-se estruturados da seguinte forma.

Capítulo **2 - Contextualização Teórica**: descreve com maiores detalhes sistemas de múltiplos modelos e introduz conceitos básicos relacionados à EES e AM.

Capítulo **3 - Revisão da Literatura**: apresenta os trabalhos realizados pela comunidade científica no desenvolvimento de métodos de AM aplicados à EES.

Capítulo **4 - Framework para seleção dinâmica de múltiplos modelos**: explana sobre o *framework* de DES proposto nesta pesquisa.

Capítulo **5 - Repositórios de dados**: descreve os repositórios de EES usados nos experimentos realizados neste trabalho.

Capítulo **6 - Resultados**: apresenta e discute os resultados dos experimentos realizados.

Capítulo **7 - Conclusão**: apresenta as considerações finais sobre os principais tópicos abordados nesta pesquisa, incluindo as contribuições alcançadas e as indicações de trabalhos futuros.

2 CONTEXTUALIZAÇÃO TEÓRICA

No presente capítulo, serão abordados temas relacionados aos Aprendizagem de Máquina (AM) com foco no de múltiplos modelos, permitindo ao leitor compreender os conteúdos apresentados neste trabalho.

2.1 FUNDAMENTOS DE AM

O principal objetivo desta seção é apresentar um entendimento geral sobre AM. Uma cobertura completa acerca do assunto pode ser encontrada em (MONARD; BARANAUSKAS, 2003; HARRINGTON, 2012; MITCHELL, 1997; WITTEN; FRANK, 2011; FACELI et al., 2011).

A AM é uma área da Inteligência Artificial (IA) que desenvolve algoritmos capazes de fazer com que o computador aprenda a partir de dados anteriores e estime comportamentos futuros. O aprendizado de máquina é capaz de identificar padrões que seriam difíceis de serem encontrados manualmente com o uso de técnicas triviais de análise de dados. A generalização é uma habilidade dos algoritmos de AM em realizar previsões com base em dados de entrada que não foram usados no treinamento (MONARD; BARANAUSKAS, 2003).

Diversas áreas de negócio utilizam-se de dados históricos e buscam soluções para problemas específicos ou para melhoria de processos existentes usando AM. Dentre diversas áreas do conhecimento, aplicações vêm sendo desenvolvidas, entre elas, medicina, educação, processamento de linguagem natural, bioinformática, detecção de fraude, reconhecimento de fala, finanças, robótica, sistemas de recomendação, mineração de textos etc. Diante do que já foi descrito sobre Estimativa de Esforço de *Software* (EES), podemos afirmar que a inteligência artificial com o uso de técnicas de AM é uma forte estratégia para se obter estimativas mais precisas e exatas em problemas de EES, assim como em problemas de regressão semelhantes.

De forma geral, existem dois tipos de aprendizagem que atuam por generalização do conhecimento: dedutiva e indutiva. A aprendizagem dedutiva utiliza-se de generalizações válidas de algo que já é conhecido para "aprender" novas informações. Usando o princípio da dedução, a aprendizagem dedutiva segue um raciocínio que parte do mais geral para o mais específico (RUSSELL; NORVIG, 2010), enquanto os métodos pertencentes à aprendizagem indutiva são mais populares e utilizados para derivar conhecimentos novos (MONARD; BARANAUSKAS, 2003). A aprendizagem indutiva, portanto, permite extrair conclusões genéricas (regras) sobre um conjunto particular de exemplos, ou seja, uma espécie de raciocínio que parte do mais específico para o geral. É o aprendizado mais desafiador, uma vez que não existe garantias que o conhecimento seja verdadeiro, o que dificulta a análise dos resultados obtidos.

O aprendizado indutivo pode ser classificado como: supervisionado ou não supervisio-

nado. No aprendizado supervisionado, a saída desejada (variável dependente) está disponível durante o treinamento, podendo ser discreta (problemas de classificação) ou contínua (problemas de regressão). No aprendizado não supervisionado, por sua vez, não existe uma variável de saída que categorize ou defina um valor contínuo para o exemplo. Neste tipo de aprendizado, é comum realizar tarefas de agrupamento ou de associação. Desta forma, tipos de aprendizados podem ser divididos nas seguintes tarefas: classificação (supervisionado), regressão (supervisionado), agrupamento (não supervisionado) e regras de associação (não supervisionado).

Antes de conceituar cada uma dessas tarefas, é preciso apresentar os diferentes tipos de dados que compõe os conjuntos de treinamento e teste. O conjunto de dados de treinamento permite que os algoritmos aprendam a partir dos dados e gerem modelos que serão avaliados em dados desconhecidos e presentes no conjunto de teste. A base de dados que permite a aprendizagem da máquina pode ser composta por dados numéricos ou categóricos, sendo que os atributos numéricos podem assumir valores reais, binários ou inteiros, enquanto os atributos categóricos são representados por um número finito de símbolos ou nomes. Ainda, os valores numéricos podem ser: (i) contínuos, quando assumem um conjunto infinito de valores, geralmente representados por valores reais; ou (ii) discretos, quando os valores são definidos por rótulos (FACELI et al., 2011).

Na classificação, existe um atributo especial chamado de *classe*. O objetivo desta tarefa é usar os atributos de entrada (variáveis independentes) que compõe o exemplo (instância), para tentar prever a *classe*, que é um valor categórico, ao passo que, na regressão, o objetivo é prever uma saída numérica. Por exemplo, prever a altura de uma pessoa a partir do peso e de outros atributos é uma tarefa de regressão, mas, em contrapartida, prever se uma pessoa está apta ou não apta a receber um cartão de crédito do banco é uma tarefa de classificação.

No aprendizado não supervisionado não existem variáveis de saída. Podemos citar como exemplo o agrupamento, que é um tipo de tarefa pertencente ao aprendizado não supervisionado. O objetivo desta tarefa é criar grupos e atribuir instâncias a estes grupos a partir das características destas instâncias. O agrupamento busca semelhanças entre as características dos próprios elementos e atribui grupos a eles (WITTEN; FRANK, 2011). Por fim, a associação busca a relação entre itens. Os algoritmos de associação geram regras para cada cenário, como, por exemplo, relacionar um cliente que comprou um produto A com a compra de um produto B, visto que uma regra foi criada porque existe uma forte relação nos dados entre as vendas do produto A com as vendas do produto B (WITTEN; FRANK, 2011). Neste trabalho será utilizado apenas o aprendizado supervisionado, também chamado de preditivo.

Uma vez compreendido os tipos de aprendizado e as respectivas tarefas, o passo seguinte é entender o que é gerado a partir de um algoritmo de AM. Vale ressaltar que um algoritmo aprende com dados históricos, logo, esse aprendizado constrói um modelo para

prever dados desconhecidos, cujos tipos variam de acordo com as famílias dos algoritmos preditivos. Um algoritmo de predição é uma função que, dado um conjunto de exemplos rotulados ou numerados na variável dependente, constrói um estimador, enquanto os rótulos tomam valores num domínio conhecido, os valores numerados pertencem ao conjunto de valores Reais. Se o domínio for um conjunto de valores nominais, tem-se um problema de classificação, e o modelo gerado é um classificador. Se o domínio for um conjunto infinito e ordenado de valores, tem-se um problema de regressão e o modelo induzido é um regressor (FACELI et al., 2011). Em um conjunto de dados, a variável dependente normalmente é representada na última coluna, a qual corresponde ao atributo alvo ou variável objetivo, em problemas de classificação, ela é designada *classe*. As colunas restantes do conjunto são designadas atributos de entrada, atributos preditivos ou variáveis independentes.

Nesse sentido, o objetivo do aprendizado preditivo é aprender uma função $f'(\mathrm{x})$ que mapeia as variáveis independentes na variável dependente. No entanto, $f'(\mathrm{x})$ é uma aproximação para uma função desconhecida $f(\mathrm{x})$. A qualidade de um modelo preditivo é dada pelo custo associado às previsões do modelo. Diferentes funções de custo podem ser usadas para avaliar os modelos. Na classificação é comum o uso da função de custo *0-1*, em que uma previsão correta resulta em *0*, e uma previsão incorreta é *1*, enquanto na regressão o *Mean Absolute Error* (MAE), *Mean Quadratic Error* (MQE) e *Median Absolute Error* (MedAE) são mais usuais. No Item 4.4.4, são apresentadas as funções de custo usadas nos experimentos desta pesquisa. Em seguida são citados diferentes grupos de algoritmos usados para induzir modelos preditivos. Nesse sentido, iremos utilizar a seguinte categorização para caracterizar os algoritmos no decorrer deste trabalho.

1. Métodos baseados em distâncias;
2. Árvores;
3. Regras de decisão;
4. Regressão simples, múltipla e logística;
5. Redes neurais;
6. Máquina de vetores de suporte
7. Métodos probabilísticos

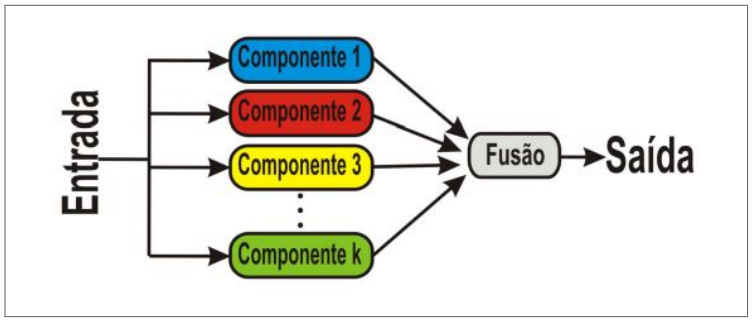
Detalhes sobre cada um desses grupos de algoritmos podem ser encontrados na literatura (MONARD; BARANAUSKAS, 2003; HARRINGTON, 2012; MITCHELL, 1997; WITTEN; FRANK, 2011; FACELI et al., 2011). No entanto, os modelos gerados a partir dos algoritmos pertencentes aos grupos acima são de natureza única, ou seja, cada um deles aprendem de maneira específica. Utilizar o aprendizado de diferentes modelos e unir o conhecimento aprendido por cada um deles forma o que chamamos de sistemas de múltiplos modelos, sendo este o assunto que abordaremos na Seção 2.2.

2.2 SISTEMAS DE MÚLTIPLOS MODELOS

Um Sistema de Múltiplos Modelos (SMM) é uma combinação de preditores em que decisões individuais são combinadas através de funções de agregação para prever novos exemplos (DIETTERICH, 1997). No entanto, combinar modelos semelhantes não tende a diminuir os erros (FACELI et al., 2011), e, por essa razão, um SMM deve conter modelos individuais diversos. Segundo (HANSEN; SALAMON, 1990), SMM são mais úteis quando os modelos individuais cometem erros independentes.

Conforme abordado em (CRUZ; SABOURIN; CAVALCANTI, 2018), o processo de construção de um SMM pode ser dividido em três etapas em que a primeira aborda a geração dos modelos. As abordagens homogêneas e heterogêneas são relacionadas com a fase de geração dos modelos, assim, utilizaremos o termo modelos bases para se referir aos modelos de um *ensemble* homogêneo, e modelos individuais, quando o contexto for de um *ensemble* heterogêneo. A segunda, denominada fase de corte, resulta na eliminação de alguns modelos. Apesar da fase de corte ser opcional, ela tem sido relatada em alguns casos para reduzir o tamanho dos conjuntos obtidos sem degradar a precisão (MOREIRA et al., 2012). Por fim, a terceira define a integração dos modelos selecionados, sendo as estratégias dessa classificadas como de fusão ou seleção. A arquitetura geral de um *ensemble* é apresentada na Figura 2.

Figura 2 – Arquitetura geral de um Ensemble



Fonte: (FACELI et al., 2011)

Pesquisadores de *Ensemble Learning* (EL) buscam soluções para duas questões: (1) como gerar um *ensemble*?; e (2) como integrar as previsões dos modelos do conjunto?

A geração dos SMM pode partir de diferentes estratégias, e seus modelos podem ser gerados a partir de subamostras da base de treinamento, oriundos de diferentes algoritmos etc. Todavia, a diversidade é um requisito importante na construção de um SMM (FACELI et al., 2011). Métodos para gerar *ensembles* homogêneos - oriundos de um mesmo algoritmo - podem ser agrupados pela maneira que geram diversidade nos classificadores bases. A amostragem de objetos (*Bagging*) ou de atributos(*Random Subspace*) são exemplos dessas diferentes maneiras de gerar *ensembles* homogêneos. Em contrapartida, os sistemas heterogêneos - formados por algoritmos distintos - garantem por natureza a diversidade

dos modelos individuais do *ensemble* (FACELI et al., 2011).

A abordagem utilizada por *ensembles* homogêneos ou heterogêneos é claramente separada da ideia de seleção de modelos. Esta estratégia seleciona os modelos de forma estática ou dinâmica, como será apresentado no Item 2.2.3. No entanto, (MOREIRA et al., 2012) mostra que abordagens de seleção não são totalmente separadas das de fusão, como geralmente é feito. De acordo com essa definição, a seleção é um caso especial de combinação em que os pesos são todos zero, exceto um ou alguns deles.

Quanto à integração das previsões dos modelos, a votação é o método de combinação mais comumente usado em problemas de classificação (FACELI et al., 2011). Ele pode ser uniforme, quando todos os classificadores contribuem igualmente para a classificação final, ou com pesos, em que cada classificador tem um peso associado podendo mudar ao longo do tempo. Além da votação, também é possível usar o método de seriação (FACELI et al., 2011), partindo de uma abordagem em que, cada classificador produz uma estimativa da probabilidade do exemplo pertencer a uma classe, ao invés de usar um único valor categórico. Essas probabilidades podem ser combinadas de diversas maneiras, como, por exemplo, no caso em que a forma mais conservadora é a soma dos valores probabilísticos. Cada modelo estima as probabilidades de uma classe ocorrer para um dado exemplo, sendo que estas probabilidades emitidas para cada classe são somadas e o exemplo é classificado de acordo com a classe que contiver o maior valor somado. Além da soma, a média, mediana, máximo e mínimo também são exemplos de combinações que podem ser adotadas.

No entanto, não existe uma definição amplamente aceita para EL direcionada a problemas de regressão (MOREIRA et al., 2012). Algumas das definições existentes são parciais, no sentido de que focam exclusivamente em problemas de classificação ou em parte do processo de aprendizagem de *ensembles* (DIETTERICH, 1997). Da mesma maneira que é utilizada para classificação, EL para regressão utiliza modelos obtidos através da aplicação de um algoritmo individual a um conjunto de dados, sendo esses modelos integrados a fim de obter a previsão final (MOREIRA et al., 2012).

Métodos de geração de modelos homogêneos têm aparecido na literatura e são amplamente utilizados até os dias de hoje. Por exemplo, *Bagging* (BREIMAN, 1996) e *Boosting* (SCHAPIRE et al., 1998) têm sido utilizados em diversos estudos (AMASAKI, 2017; AZZEH et al., 2018; MALGONDE; CHARI, 2019; ABDELALI; MUSTAPHA; ABDELWAHED, 2019; RAO; RAO, 2020; SINGHAL, 2020). Quantos aos heterogêneos, o *Stacking* (WOLPERT, 1992) é o método mais comum, combinando modelos formados por diferentes algoritmos através de um meta modelo que utiliza as saídas dos modelos individuais para prever a saída final. *Ensembles* heterogêneos também continuam sendo usados na literatura (HOSNI; IDRI; ABRAN, 2017; HOSNI et al., 2017; HOSNI; IDRI; ABRAN, 2018; SHAH et al., 2020; MALGONDE; CHARI, 2019; PALANISWAMY; VENKATESAN, 2020). Descreveremos nos itens 2.2.1 e 2.2.2, respectivamente, os métodos usados para gerar *ensembles* homogêneos e heterogêneos que

foram utilizados como *baselines* nos experimentos realizados neste trabalho.

2.2.1 *Ensembles* homogêneos

SMM baseados no mesmo algoritmo são mais comuns na literatura de EES (IDRI; HOSNI; ABRAN, 2016a). Os métodos de geração de *ensembles* homogêneos mais usados são: *Bagging* e *Boosting*, os quais são baseados na amostragem dos exemplos de treinamento e são descritos a seguir.

1. *Bagging* (*Bootstrap AGGreatING*):

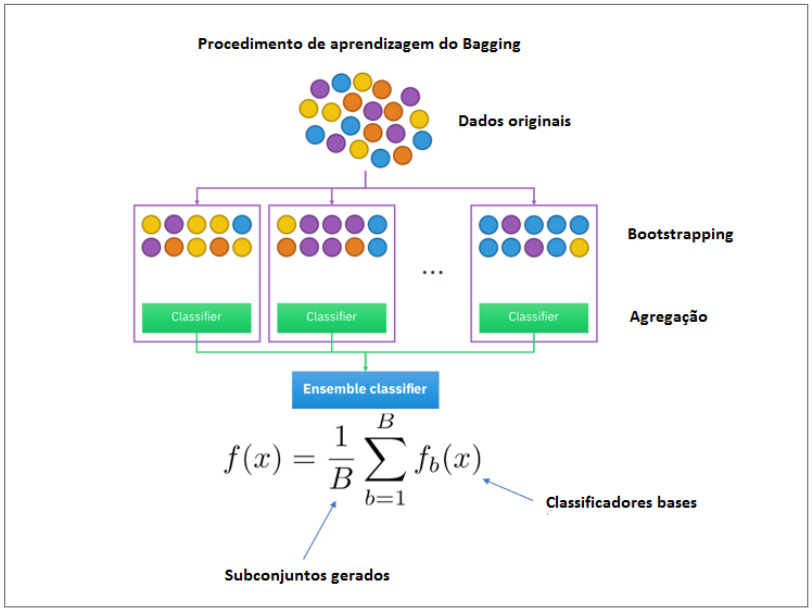
   O *Bagging* é um método proposto por (BREIMAN, 1996), baseado na amostragem *bootstrap* (GRIMSHAW, 1995), bastante conhecido e utilizado na literatura de AM. O método gera diversos modelos a partir dos dados de treinamento, pois, uma grande quantidade de subconjuntos de dados de mesmo tamanho são extraídos n vezes. Os exemplos são selecionados aleatoriamente e colocados novamente no conjunto de dados, o que caracteriza uma amostragem com reposição. Isso leva a alguns exemplos não aparecerem na amostra, enquanto outros podem aparecer mais de uma vez. Os subconjuntos gerados são utilizados para treinar os modelos bases, consequentemente, o procedimento resulta em n modelos de AM que terão as suas saídas agregadas posteriormente. Em domínios de classificação, a integração dos modelos normalmente é pelo voto uniforme. Por outro lado, em problemas de regressão, a saída final é dada pela média. Este algoritmo é geralmente usado para aumentar o desempenho e diminuir a variabilidade aleatória dos modelos bases (FACELI et al., 2011).

   A Figura 3 apresenta o processo de aprendizagem do *Bagging*. Perceba que β subconjuntos são gerados a partir dos dados originais, em seguida, os modelos são criados e agregados em um esquema de combinação paralela. O exemplo da Figura 3 usou a média como regra de combinação e, nesse sentido, estamos diante de um problema de regressão.

2. *Boosting*:

   Assim como o *bagging*, o *Boosting*, cujos detalhes podem ser encontrados em (SCHAPIRE et al., 1998) utiliza o mesmo algoritmo para gerar n modelos. No entanto, ele lança mão de diferentes distribuições do conjunto de treinamento para combinar as saídas dos modelos. A mudança na distribuição dos dados é baseada na análise de erros do classificador anterior, fazendo com que esta característica o torne diferente do *Bagging*, que é um método caracterizado por ser sequencial. O *Boosting* busca melhorar o desempenho de cada um dos novos classificadores, levando o método a aprimorar os modelos bases gradativamente. Nesse sentido, ele é uma técnica que combina algoritmos "fracos"com o objetivo de aperfeiçoar os resultados, corrigindo

Figura 3 – Processo de aprendizagem do algoritmo de *Bagging*



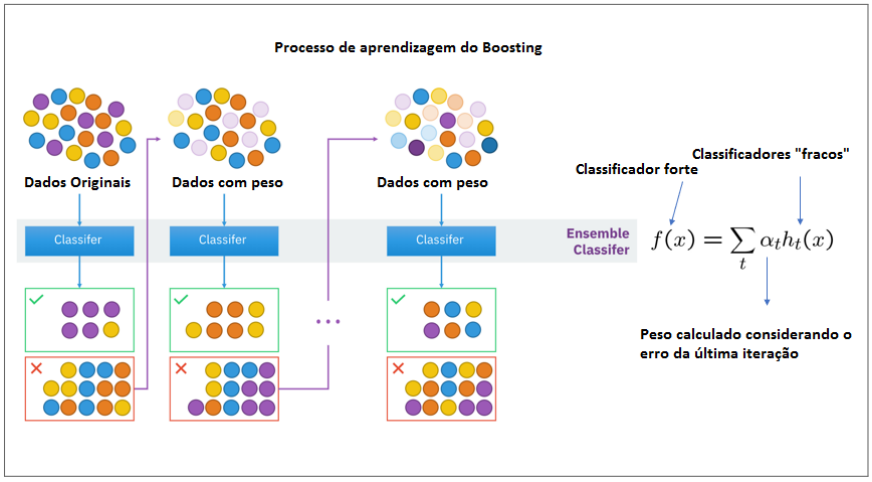
Fonte: (SINGHAL, 2020)

assim os erros dos modelos individuais. O estudo (SCHAPIRE et al., 1998) mostrou que um classificador forte pode ser gerado a partir de classificadores fracos. Um algoritmo de adaptação bastante usado e conhecido na literatura é o *AdaBoost* (*Adaptive Boosting*), que foi apresentado por Freund e Schapire em (TORRES-SOSPEDRA; HERNÁNDEZ-ESPINOSA; FERNÁNDEZ-REDONDO, 2006) e também foi utilizado nos experimentos realizados neste trabalho.

A Figura 4 apresenta o processo de aprendizagem do *Boosting*. Perceba que a ideia do algoritmo é associar um peso para cada exemplo no conjunto de treinamento, refletindo assim a importância de cada instância. Em cada iteração é gerado um novo modelo, que é treinado com a distribuição dos exemplos dados pelos pesos associados. Os pesos são associados de acordo com o desempenho do conjunto de modelos aprendidos até essa iteração. Podemos notar que na segunda iteração da Figura 4, os exemplos estimados incorretamente na iteração anterior receberam um peso maior (cores intensas), enquanto os que foram estimados corretamente tiveram menos importância posteriormente, portanto receberam pesos menores. O procedimento continua até que θ modelos sejam treinados. O modelo final agrega os modelos aprendidos em cada iteração pela votação ou média ponderada, sendo que o peso de cada modelo é uma função de sua precisão (FACELI et al., 2011).

Bagging e *Boosting* não são os únicos métodos usados para geração de modelos homogêneos, também existem métodos que são baseados na amostragem de atributos (*Random Subspace* (HO, 1998a)), na injeção de aleatoriedade, que consiste em inicializar hiper parâmetros do algoritmo aleatoriamente como, por exemplo, iniciar os pesos de uma rede

Figura 4 – Processo de aprendizagem do algoritmo de *Boosting*



Fonte: (SINGHAL, 2020)

neural com diferentes valores (FACELI et al., 2011), entre outros. Além destes métodos, outra abordagem, introduzida por Breiman (BREIMAN, 2001) apresentou um algoritmo de florestas aleatórias, do inglês - *Random Florest*. Em síntese, o modelo cria diferentes árvores de decisão a partir da amostragem de exemplos com reposição e da seleção aleatória de atributos, combinando assim, as decisões por votação uniforme. O *Random Forest* foi um dos algoritmos avaliados no processo de seleção dinâmica dos modelos de regressão.

Neste trabalho, os métodos *Bagging* e *Boosting* foram investigados, a fim de comparar e avaliar os resultados alcançados por eles e por outros métodos, incluindo obviamente os métodos gerados a partir do *framework* proposto.

2.2.2 *Ensembles* heterogêneos

Um sistema de múltiplos modelos heterogêneos utiliza diferentes algoritmos individuais, garantindo assim a diversidade entre modelos. O método *Stacking* é conhecido na literatura como uma generalização empilhada de modelos (WOLPERT, 1992), que busca saber qual a melhor forma de combinar as saídas dos modelos individuais (SHUNMUGAPRIYA; KANMANI, 2013). Nesta estratégia de modelos empilhados, as saídas são combinadas por um meta-classificador que gera uma estimativa derivada das previsões individuais.

O aprendizado do *Stacking* é baseado normalmente em duas camadas, mas o método pode ter várias. Na camada mais baixa estão os modelos de Nível$_0$, que recebem como entrada os dados originais, de tal forma que, cada modelo desse nível produz uma predição. A camada sucessiva à anterior é a camada Nível$_1$, que recebe como entrada as predições da camada precedente e, em seguida, a saída é passada para a camada posterior. O processo continua até que o modelo de mais alto nível realize a predição, no entanto, assim como já foi afirmado, a maioria dos trabalhos que envolvem este método concentra-se em duas camadas (FACELI et al., 2011).

O processo de aprendizado usado pelo algoritmo consiste nos seguintes passos:

1. Treinar os classificadores de Nível_0

   Os classificadores são treinados através de um processo de validação *leave-one-out*, que será discutido em mais detalhes no Item 4.4.3. Nesse processo um elemento do conjunto é deixado de fora e os demais elementos são usados para o treinamento do modelo. Depois que todos os exemplos foram separados para avaliação, o modelo realiza a predição de cada exemplo que foi excluído. Em seguida, é criado um vetor de probabilidades a partir das predições dos modelos de Nível_0 e das classes atuais dos exemplos, em que cada classificador registra a probabilidade de uma classe para um exemplo. Um esquema com três classificadores no Nível_0 e um problema binário levará a um vetor com seis probabilidades (2 classes x 3 modelos) e uma classe de saída.

2. Treinar o classificador Nível_1

   A partir do conjunto de dados gerados pelos vetores da fase anterior, um classificador é treinado. O número de exemplos dos dados do Nível_1 é igual ao número de exemplos do conjunto original, no entanto, a quantidade de atributos será o produto do número de modelos do nível anterior pelo número de classes do domínio. O meta-classificador criado vai ser usado para estimar a saída de um exemplo de teste.

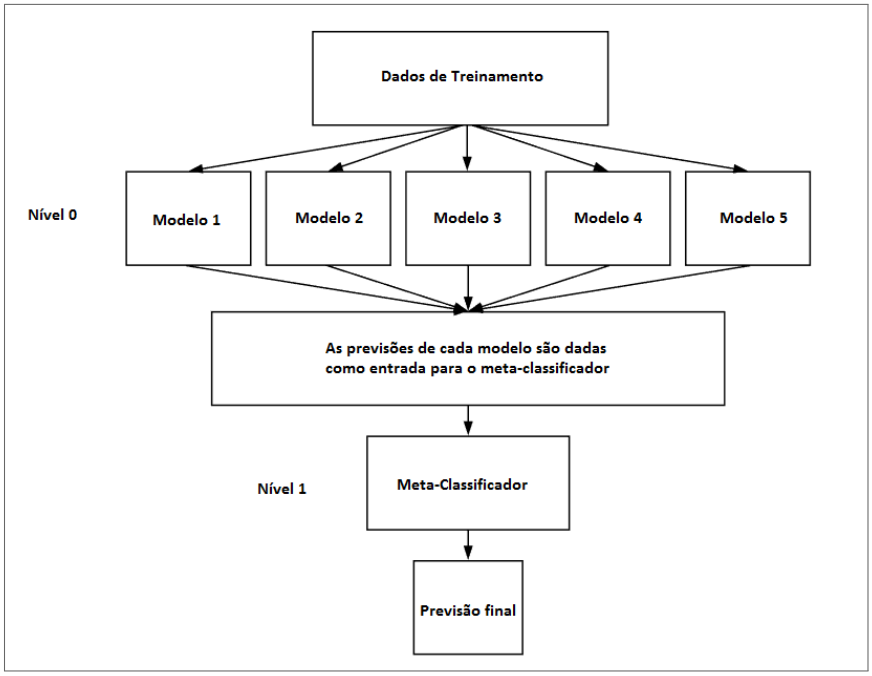
3. Retreinar classificadores

   No passo 1, os modelos são gerados deixando um elemento de fora. Para utilizar todo o conjunto de treinamento, os classificadores do Nível_0 são retreinados, usando o conjunto completo. Os modelos gerados são usados para classificar os exemplos de teste, sendo que cada probabilidade informada pelos modelos do nível anterior é passada como entrada para o meta-classificador e, este então realiza a estimativa final do sistema.

Na aplicação do algoritmo, os modelos do Nível_0 classificam o exemplo de teste por probabilidades, gerando o vetor de predições que é enviado ao classificador Nível_1, responsável por produzir a saída para um esquema de generalização em pilha com duas camadas. A Figura 5 representa o esquema do *Stacking*, e na Tabela 2.2.2 é dado um exemplo com 5 atributos de entrada e 1 classe de saída binária representando os dados originais nas seis primeiras colunas. Nas colunas seguintes, P_{ik} representa a probabilidade dada pelo classificador i para a classe k, a mesma do conjunto de dados original. O classificador de Nível_1 utiliza o conjunto de dados com as probabilidades para classificar os exemplos novos. O *Stacking* é uma técnica sofisticada para reduzir o erro devido à redução do viés provocado pelos modelos individuais (WOLPERT, 1992).

Outra maneira simples e eficaz de combinar modelos heterogêneos é simplesmente agregando as saídas com operações lineares (média, mediana etc.). Em (POSPIESZNY;

Figura 5 – Esquema de funcionamento do *Stacking*



Fonte:(FACELI et al., 2011)

Tabela 1 – Exemplo do método de *Stacking*

A1	**A2**	**A3**	**A4**	**A5**	**Classe**	P_{11}	P_{12}	P_{21}	P_{22}	P_{31}	P_{32}	**Classe**
t	a	c	t	a	Membro	0,51	0,49	0,13	0,87	0,12	0,88	Membro
t	g	c	t	a	Membro	0,19	0,81	0,07	0,93	0,81	0,19	Membro
g	t	a	c	t	Não Membro	0,68	0,32	0,55	0,45	0,69	0,31	Não Membro
a	a	t	t	g	Membro	0,74	0,26	0,66	0,34	0,94	0,06	Membro
t	c	g	a	t	Não Membro	0,62	0,38	0,01	0,99	0,78	0,22	Não Membro
a	g	g	g	g	Membro	0,65	0,35	0,90	0,10	0,55	0,45	Membro

Fonte: O autor (2022)

CZARNACKA-CHROBOT; KOBYLINSKI, 2018) foi apresentada uma combinação de *Support Vector Regression* (SVR), *Multi Layer Perceptron* (MLP) e *Multi Linear Regression* (MLR). Os resultados da pesquisa mostraram que a média como função de combinação dos modelos levou a resultados precisos. A acurácia de modelos agregados com simples regras de combinação em EES tem indicado que os *ensembles* são mais precisos do que os métodos individuais (IDRI; HOSNI; ABRAN, 2016a). Além disso, estes métodos são facilmente implementados na prática. Esta agregação também pode ser realizada através de operações não lineares como, por exemplo, utilizando um meta-modelo, semelhante à estratégia usada pelo *Stacking* que recebe como entrada a saída dos modelos individuais e dá como uma saída a estimativa final.

2.2.3 Métodos de seleção de modelos

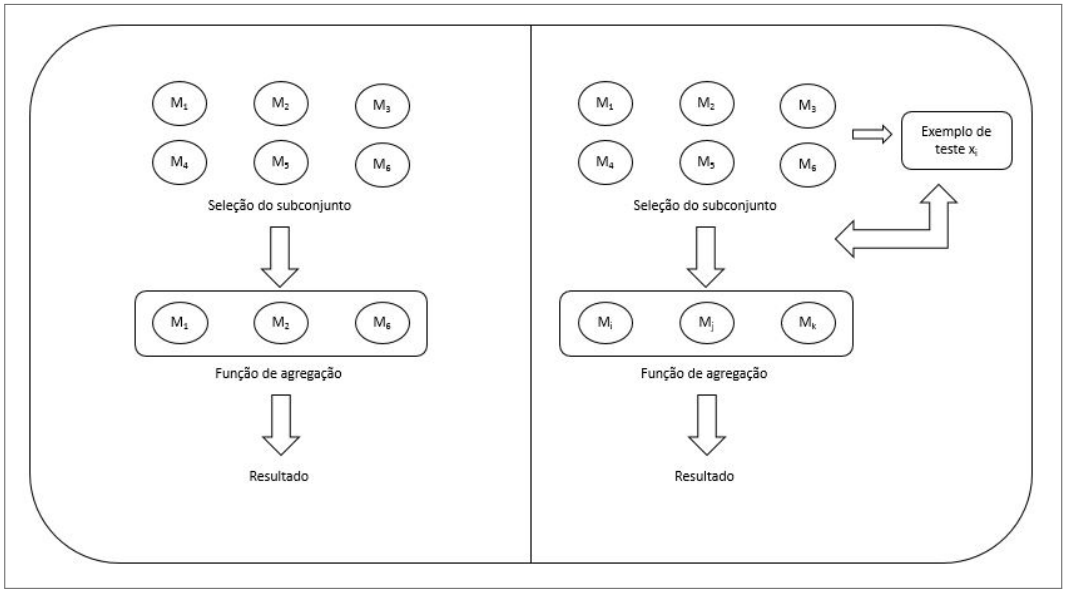
A distribuição da taxa de erro sobre o espaço de atributos geralmente não é homogênea. Dependendo do modelo, a taxa de erro será mais concentrada em certas regiões do espaço de objetos (FACELI et al., 2011). A escolha de um algoritmo para um dado problema se dá basicamente pelo tipo de domínio e pelas características dos algoritmos, no entanto avaliações prévias são recomendadas antes do modelo selecionado entrar em produção para realizar estimativas de novos exemplos. A abordagem mais comum dessas avaliações é dividir os dados em três partes: (1) o conjunto de treinamento, usado para obter os preditores bases; (2) o conjunto de validação, utilizado para avaliação do erro de validação, que representa a capacidade de generalização dos preditores bases; e (3) o conjunto de teste, usado para avaliar o conjunto final. Todavia, neste esquema, o modelo ou o conjunto de modelos escolhidos após a avaliação são selecionados para prever todos os exemplos novos, o que caracteriza uma seleção estática. Nesta seleção, todos os dados de teste são previstos com o mesmo subconjunto de modelos.

Entretanto, é comumente aceito que as instâncias de testes estejam geralmente relacionadas a diferentes níveis de dificuldades de previsão, o que leva muitos pesquisadores a investigarem o uso da seleção dinâmica em diversos tipos de problemas. Nesse tipo de seleção, a amostra de teste é prevista por um modelo ou a partir de um subconjunto de modelos que são definidos de acordo com as características do espaço dimensional. Nesse sentido, se na seleção estática todos os modelos selecionados antecipadamente são usados para prever todas as amostras de teste, na seleção dinâmica devem ser selecionados apenas os modelos mais apropriados para cada amostra de teste.

Técnicas de seleção dinâmica buscam encontrar os melhores modelos para diferentes regiões dentro do domínio do problema e, tendem a obter menor erro nas previsões do que a seleção estática, pois, pesquisas recentes trazem resultados em que elas superaram a seleção estática (KO; SABOURIN; BRITTO JR., 2008; BRITTO; SABOURIN; OLIVEIRA, 2014; CRUZ; SABOURIN; CAVALCANTI, 2018). Além disso, estudos da literatura mostram que ótimos resultados podem ser alcançados selecionando um subconjunto de modelos ao invés de um único modelo para cada instância de teste (KO; SABOURIN; BRITTO JR., 2008). Normalmente, os modelos são selecionados com base nas características ou regiões espaciais de cada instância.

Na Figura 6, são apresentadas duas formas de seleção de subconjuntos de modelos. Observe que ela mostra os modelos 1, 5 e 6 sendo usados em todas as instâncias de teste, o que caracteriza uma seleção estática, enquanto na seleção dinâmica os modelos selecionados variaram de acordo com as características do exemplo de teste x. Dessa forma, diferentes instâncias de teste podem ser previstas por diferentes subconjuntos de modelos. Usaremos neste trabalho o termo *Dynamic Selection* (DS) para se referir a seleção dinâmica de um único modelo, e *Dynamic Ensemble Selection* (DES) para a seleção dinâmica de vários modelos.

Figura 6 – Esquemas de seleção de modelos



Fonte: (KO; SABOURIN; BRITTO JR., 2008)

A seleção dinâmica traz vantagens que nos leva a investigá-la diante da seleção estática. Essa avaliação pode ser iniciada pela acurácia local dos modelos, que é o critério mais comum utilizado pelas técnicas de seleção dinâmica. A seguir são apresentados os métodos mais usados na literatura de seleção dinâmica de acordo com o estudo (BRITTO; SABOURIN; OLIVEIRA, 2014). O primeiro é o *Dynamic Classifier Selection* (DCS) que realiza a seleção dinâmica de um modelo de classificação, e o segundo é o *K-Nearest Oracle* (KNORA), que seleciona um subconjunto de modelos dinamicamente.

2.2.4 Seleção dinâmica de um modelo

As principais estratégias de seleção dinâmica são baseadas na taxa de acerto ou precisão local de cada modelo.

O DCS é uma abordagem para a seleção dinâmica de um classificador baseada na acurácia da estimativa local. Desenvolvido para problemas de classificação, o método estima a precisão de cada classificador em regiões locais do espaço de características, ou seja, ao redor da amostra de teste desconhecida. Portanto, o classificador mais preciso localmente é selecionado. O Algoritmo 1 descreve o método. Ainda, (WOODS; KEGELMEYER; BOWYER, 1997) propuseram dois métodos para estimar a taxa de acerto local: *Overall Local Accuracy* (OLA) e *Local Class Accuracy* (LCA), como mostram as Figuras 7 e 8 que apresentam um exemplo do uso destes métodos. O OLA utiliza a abordagem de *rankings* do classificador, que seleciona o modelo com melhor acurácia na região local. O LCA é um método semelhante ao OLA, mas ao invés da acurácia local, é usada a precisão local estimada em relação à classe de saída. Por fim, o *Dynamic Classifier Selection By Local Accuracy Weighted* (DCS-LAW) é uma abordagem que utiliza pesos, ou seja, é atribuído

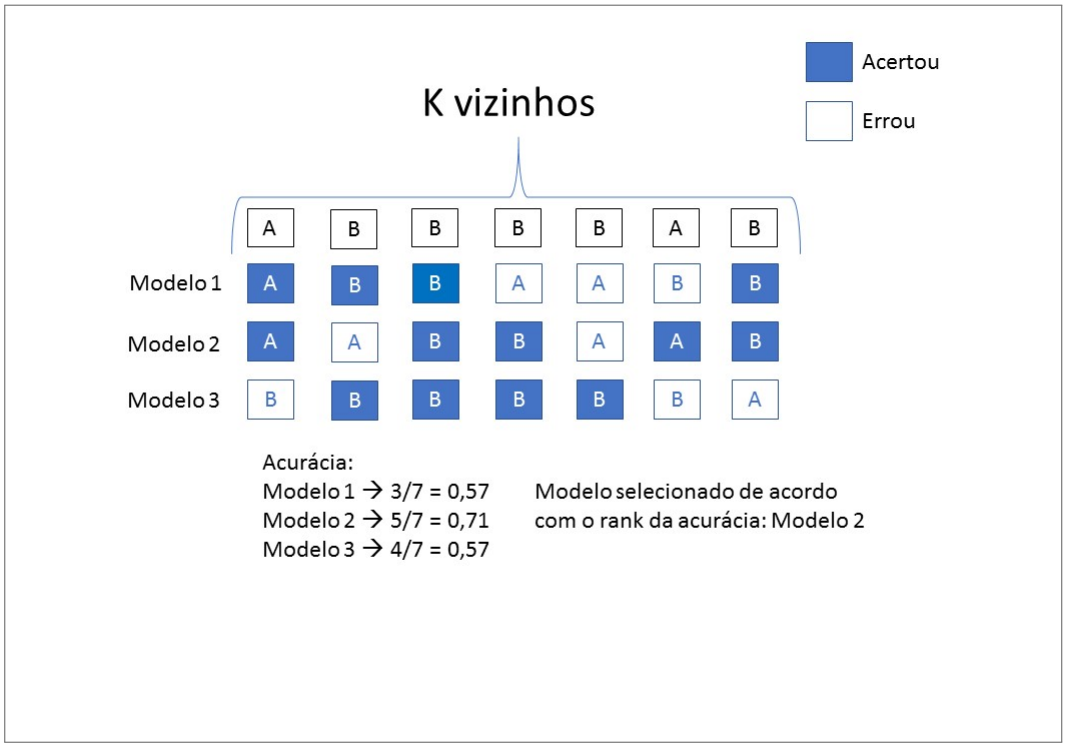
um peso a cada modelo base, de acordo com seu desempenho ao redor da amostra de teste, considerando a base de validação.

Algoritmo 1: *Dynamic Classifier Selection By Local Accuracy* (DCS-LA)

Entrada: Conjunto de Treinamento $T = (X_i, y_i), i = 1, ..., n$;

Pool de classificadores C de tamanho M; Parâmetro K vizinhos mais próximos
para $teste_i \in TESTE$ **faça**
  R_{teste_i} = região de amostras locais ao redor de $teste_i$ em T com K instâncias
  $C_{selecionado}$ = Acurácia 0
  enquanto Existir classificador no pool C **faça**
    OLA = Acurácia local (C_i, R_{teste_i});
    se C_i é mais acurado do que $C_{selecionado}$ **então**
      $C_{selecionado} = C_i$;
    fim do se
  fim do enquanto
  Classificar $Teste_i$ com o classificador selecionado $C_{selecionado}$
fim do para

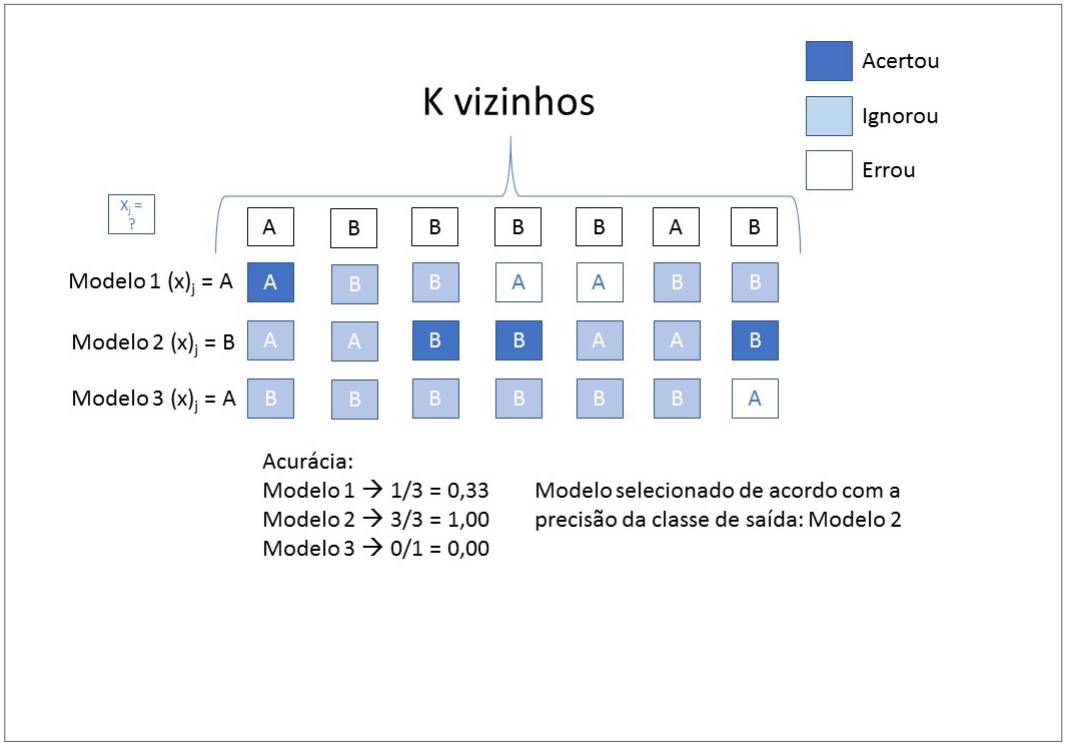
Figura 7 – Seleção dinâmica de um modelo de classificação por acurácia



Fonte: O autor (2022)

Podemos notar que nos exemplos apresentados nas Figuras 7 e 8, o modelo 2 foi selecionado para os dois métodos. No entanto, é comum que os modelos selecionados sejam distintos, uma vez que os valores da acurácia e da precisão podem ser diferentes. Na Figura 7, o OLA faz um *ranking* dos modelos de classificação, aquele com maior porcentagem de acertos dos vizinhos mais próximos é selecionado. Um parâmetro importante nesse

Figura 8 – Seleção dinâmica de um modelo de classificação pela precisão da classe de saída.



Fonte: O autor (2022)

método é o valor de k, porém, no exemplo $k = 7$, se o valor fosse $k = 5$, o modelo 3 seria o selecionado, e se tivéssemos $k = 3$, o modelo 1 que seria selecionado. Em consequência, o valor de k pode influenciar nos resultados.

Na Figura 8, o LCA é usado para avaliar a quantidade de acerto da classe de saída de cada modelo para o padrão x_j. As precisões locais são calculadas baseadas na classe atribuída a x_j por cada classificador. Esses métodos são normalmente usados em problemas de classificação, porém, para regressão, o OLA é facilmente adaptado, já que, ao invés da taxa de acertos, pode-se utilizar qualquer uma das métricas abordadas na Seção 4.4.4.

2.2.5 Seleção dinâmica de múltiplos modelos

O processo de DES considera que mais de um modelo é selecionado para prever o valor esperado de cada instância de teste. O KNORA é um método de seleção dinâmica de múltiplos classificadores, conforme é detalhado em (KO; SABOURIN; JR., 2007). Esse método usa a topologia em paralelo para agregar as estimativas, considerando que todos os classificadores podem responder ao mesmo problema de classificação. Nesse sentido, o resultado obtido pela técnica é direcionado a um único ponto.

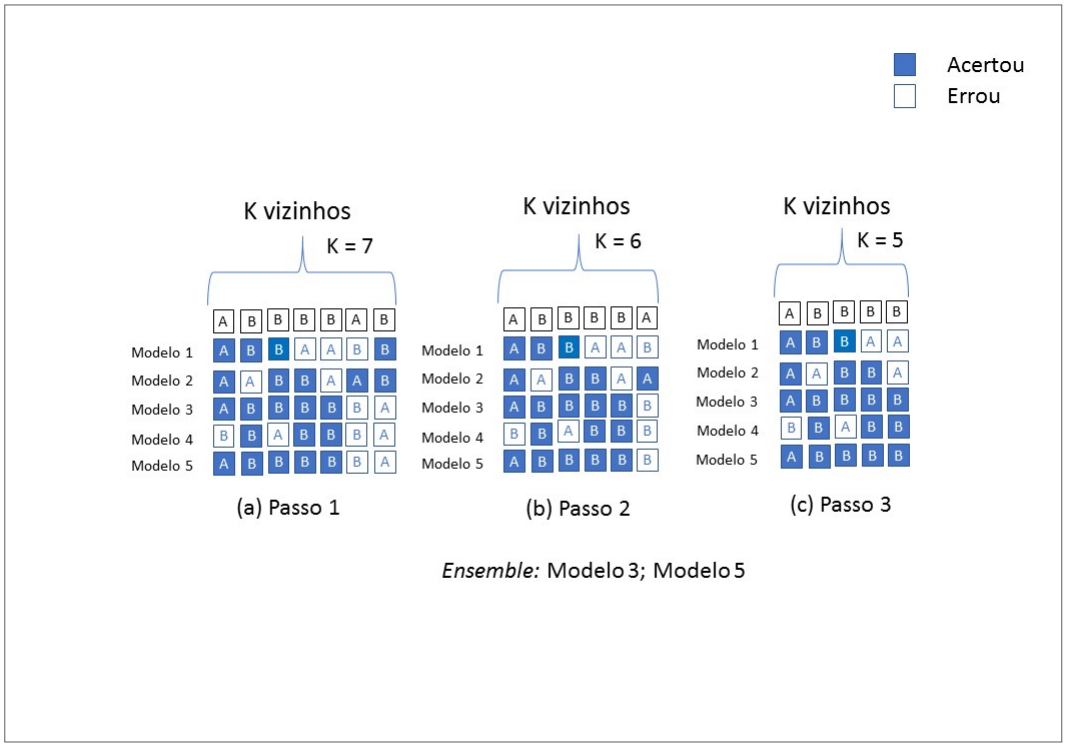
O KNORA utiliza o conceito de oráculo, o qual está ligado à seleção dos melhores subconjuntos de modelos (KUNCHEVA, 2002), cujo objetivo é encontrar os modelos de melhores desempenho para cada instância. A seleção usada pelo KNORA é baseada nas melhores estimativas alcançadas nos elementos k-vizinhos mais próximos à instância de

teste que estejam contidos na base de validação, lembrando que estes elementos compõem a região de competência. Para cada instância de teste i, o KNORA armazena o classificador que acertou as classes dos vizinhos de i, ou seja, isso quer dizer que o classificador base é selecionado se estimar corretamente a classe dos k vizinhos mais próximos de i. Porém, o KNORA utiliza diferentes estratégias para a seleção dos modelos bases. O método foi proposto em vários esquemas diferentes. A seguir será abordado os principais.

K-Nearest Oracle Eliminate (KNORA-E): seleciona o subconjunto de classificadores que estimou corretamente todos os k-vizinhos mais próximos e, nesse sentido, o modelo que não atingir o critério é eliminado. Quando nenhum classificador atinge este nível de classificação, a instância de validação mais distante do ponto de teste é retirada do grupo de vizinhos mais próximos e o processo se repete até que pelo menos um dos classificadores do *pool* acerte todos os exemplos de validação. Os classificadores que acertarem toda a vizinhança são adicionados ao *ensemble*, de modo que a cada classificador inserido no *ensemble* é dado um voto. O *K-Nearest Oracle Eliminate - Weighted* (KNORA-ELIMINATE-W) é uma extensão do KNORA-E, em que cada voto tem um peso baseado na distância entre as instâncias de validação e a instância de teste. A Figura 9 apresenta um exemplo do KNORA-E para um problema com 5 modelos de classificação e uma região de competência com 7 vizinhos. A Figura 9(a) apresenta o resultado referente aos 7 vizinhos mais próximos, de forma que podemos notar que nenhum modelo acertou a classe de todos os elementos. A região de competência neste caso é decrementada. Na Figura 9(b), tem-se a nova região de competência, mas ainda não é encontrado um modelo ótimo. Finalmente, na Figura 9(c), o método seleciona os modelos 3 e 5, que, de acordo com o exemplo, conseguiram acertar todas as instâncias de validação, quando k chegou ao valor $k = 5$.

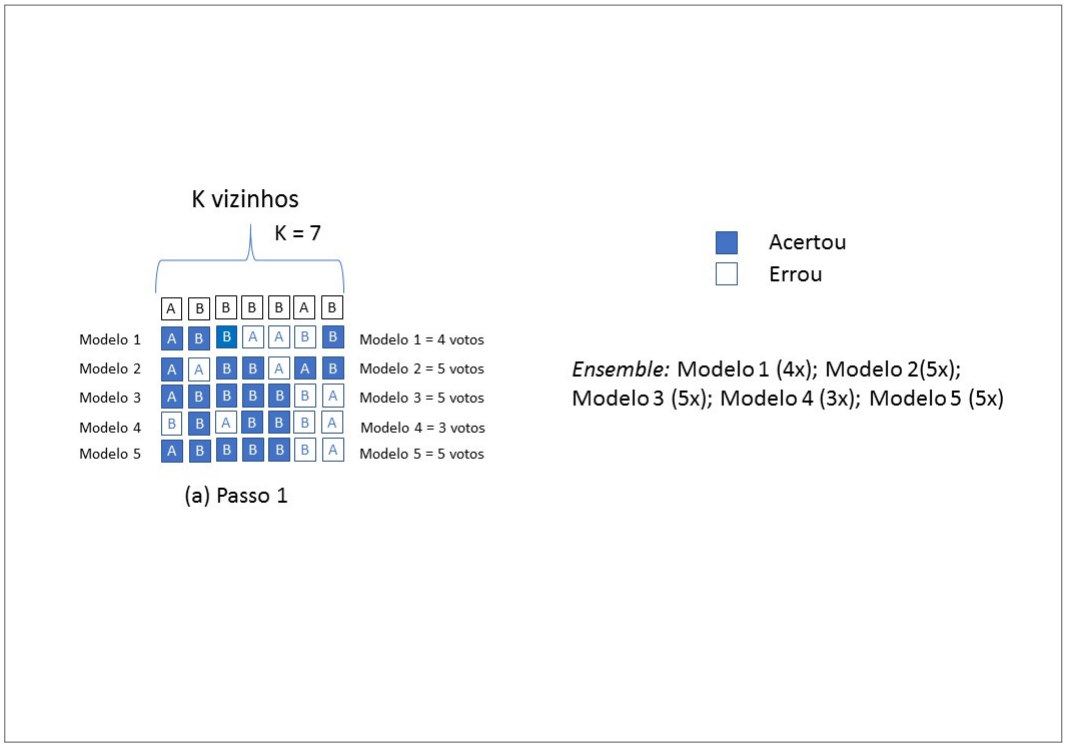
K-Nearest Oracle Union (KNORA-U): seleciona todos os classificadores que corresponderam a pelo menos um dos vizinhos mais próximos; se o classificador classificar corretamente mais de um vizinho, terá a quantidade de votos proporcional à quantidade de instâncias corretas para aquela instância de teste. O *K-Nearest Oracle Union - Weighted* (KNORA-UNION-W) é uma extensão do KNORA-U, mas cada voto tem um peso baseado na distância entre as instâncias de validação e a instância de teste. A Figura 10 apresenta um exemplo do KNORA-U em que todos os classificadores foram selecionados com os seus respectivos votos.

Figura 9 – KNORA-E



Fonte: O autor (2022)

Figura 10 – KNORA-U



Fonte: O autor (2022)

3 REVISÃO DA LITERATURA

Neste Capítulo, serão apresentados estudos relacionados ao uso de sistemas de múltiplos modelos e aplicação destes métodos em problemas de Estimativa de Esforço de *Software* (EES) e *Education Data Mining* (EDM).

3.1 TRABALHOS RELACIONADOS A SISTEMAS DE MÚLTIPLOS MODELOS

Os estudos que abordam sistemas de múltiplos modelos normalmente focam no tipo de ensemble (homogêneo ou heterogêneo), na estratégia de seleção (estática ou dinâmica), no tipo de problema abordado (classificação ou regressão) e nos critérios de seleção usados para definir os modelos adequados. Neste sentido, a seguir, serão citados vários métodos de seleção estática ou dinâmica que utilizam diferentes critérios de avaliação e que geram modelos homogêneos ou heterogêneos para problemas de classificação ou regressão.

Os *ensembles* homogêneos manipulam o conjunto de treinamento para gerar múltiplas hipóteses. O desempenho destes métodos aumenta quando o algoritmo base utilizado é instável (FACELI et al., 2011), ou seja, algoritmos cuja saída do modelo sofre mudanças em resposta a pequenas alterações nos dados de treinamento. A forma de buscar diminuir a instabilidade do algoritmo base é através da diversidade dos modelos bases. Neste sentido, os ensembles homogêneos geram diversidade através de amostragem dos exemplos, amostragem de atributos, injeção de aleatoriedade, alteração dos hiper parâmetros do algoritmo ou pela combinação dessas estratégias.

O *Bagging* (BREIMAN, 1996) e o *Boosting* (SCHAPIRE et al., 1998) são os métodos de geração de *ensembles* homogêneos mais comuns na literatura, os quais já foram citados no Capítulo 2. O *Random Forest*, introduzido por (HO, 1995), utiliza a amostragem aleatória de atributos (HO, 1998a). No entanto, posteriormente ele foi atualizado por (BREIMAN, 2001), propondo a combinação da mesma ideia implementada no *Bagging*, com adição da seleção aleatória de atributos, enquanto o *Stacking* foi sugerido em (WOLPERT, 1992). Este método é categorizado como um *ensemble* heterogêneo composto por um arquiterura de aprendizado em camadas. Apesar de todos estes métodos serem classificados como sistemas de múltiplos modelos, é comum encontrá-los como sendo parte dos métodos de Sistema de Múltiplos Modelos (SMM), ora para gerar modelos homogêneos que podem ser selecionados de forma estática ou dinâmica, ora sendo integrante de um *ensemble* heterogêneo. Assim como os dois *ensembles* homogêneos usados como métodos de avaliação neste trabalho, o *Stacking* também foi abordado com mais detalhes no Capítulo 2.

O primeiro método de seleção estática tratado nesta seção foi o *Basic Ensemble Method* (BEM), o qual é um método básico para combinação de modelos homogêneos. Neste, a combinação dos modelos é realizada através da média das saídas dos modelos bases, de

forma que todos dos modelos terão a mesma importância, conforme definiu (PERRONE; COOPER, 1993). O uso de um BEM pode reduzir o erro, quando comparado à performance de um modelo base. Segundo (PERRONE; COOPER, 1993), quanto maior o tamanho da base de dados (N), menor tende a ser o erro do BEM. No entanto, a melhoria pode ocorrer até um valor limite de N, que é quando o método não consegue aumentar o seu desempenho. O BEM pode ser visto como um regularizador que suaviza o espaço funcional em busca da verdadeira função de regressão, cujo benefício adicional é a flexibilidade adquirida, uma vez que as estimativas de regressão podem vir de muitas fontes diferentes.

O segundo método de seleção estática apresentado por (PERRONE; COOPER, 1993) foi o Generalized Ensemble Method (GEM), que apresenta modelos combinados por meio da ponderação dos modelos bases. Isto significa que os pesos atribuídos são inversamente proporcionais ao desempenho dos modelos no conjunto de treinamento ou no conjunto de validação. Desta forma, os pesos atribuídos são constantes, ou seja, não se alteram para cada instância de teste. Os *ensembles* homogêneos avaliados neste trabalho não foram ponderados. Utilizar pesos para cada modelo pode ser visto como uma estratégia de seleção estática, desde que os modelos individuais tenham sido selecionados anteriormente por meio de um processo de validação. No entanto, uma vez que o GEM considerou os pesos dos modelos a partir do desempenho deles na fase de treinamento e validação, o método pode ser categorizado como uma seleção estática de *ensemble*. O grau de importância de cada modelo individual na saída final é apresentada em (PERRONE; COOPER, 1993), e, assim, os dois métodos se diferenciam basicamente pelo peso atribuído aos modelos bases, entretanto, ambos são métodos de seleção estática de *ensembles* homogêneos, uma vez que os pesos não se alteram para as diferentes instâncias e os modelos bases são oriundos do mesmo algoritmo.

Apesar dos avanços alcançados através do uso de *ensembles*, testes iniciais foram realizados com as bases de dados usadas neste trabalho, tendo sido verificado que utilizar a fusão das saídas de muitos modelos através de um *ensemble* básico poderá diminuir o desempenho final do *ensemble*, cuja alteração pode vir da quantidade de modelos usados no *ensemble*. Isto pode ser explicado pelo motivo de existir maior probabilidade de modelos com baixo desempenho, contribuindo para a estimativa final do *ensemble*. Selecionar modelos heterogêneos de baixo desempenho pode enfraquecer o desempenho do método combinado. Neste sentido, a quantidade de modelos individuais e a combinação de modelos ideal podem ser vistos como um processo de otimização do método. Desta forma, é comum tentar selecionar um subconjunto de modelos ideal a partir de um conjunto maior. Esta estratégia gera um *ensemble* estático mais simples do que o conjunto inicial, uma vez que a quantidade de modelos é menor. Entretanto, a definição da quantidade de modelos e a seleção de modelos ideal deve ser baseada na performance dos modelos na validação, sendo validados tanto de maneira individual quanto combinada.

No entanto, definir o melhor subconjunto é uma tarefa difícil, ao menos, é necessário

tempo e recursos por parte dos pesquisadores. No estudo de (PARTALAS et al., 2008) foi usada uma busca gulosa (*greedy search*) para selecionar o melhor subconjunto de regressores baseado no desempenho dos modelos no conjunto de validação, tendo sido avaliados dois tipos de algoritmos para definir o melhor *ensemble*, de tal forma que a seleção foi estática porque todos os subconjuntos de modelos foram os mesmos para todas as instâncias de teste. Os ensembles gerados usaram algoritmos de Redes Neurais Artificiais (RNA) e *Support Vector Machine* (SVM). O primeiro método de seleção dos modelos foi baseado em *Forward Selection*, que inicia sem nenhum modelo e inclui um por vez. Este método alcançou melhor desempenho do que o segundo, o qual utilizou a estratégia de *Backward Selection*, que, ao contrário da primeiro, inicia com todos os modelos e remove um por vez. Conforme dito, a depender da quantidade de modelos bases, a busca pelo melhor subconjunto pode ser custosa e inviável, a depender do problema. Neste trabalho, testes de hipóteses deram suporte a seleção dos modelos heterogêneos ideais para cada conjunto de dados.

Um dos primeiros sistemas dinâmicos de seleção de modelos foi proposto por (ROONEY et al., 2004). Os autores propuseram três algoritmos de seleção dinâmica de regressores. Os modelos de regressão usados no *ensemble* são homogêneos e utilizam como critério de seleção o erro acumulado na região de competência. Outrossim, foram investigados dois algoritmos no estudo, uma regressão linear e um *K-Nearest Neighbor* (KNN). Cada um dos algoritmos geraram modelos homogêneos a partir da seleção de subespaços aleatórios (HO, 1998b), sendo que três técnicas foram propostas neste estudo, conforme os parágrafos em que cada uma delas será abordada.

A primeira técnica é a *Dynamic Selection* (DS), que faz uma seleção dinâmica simples e escolhe o regressor com o menor erro acumulado na região de competência. O erro acumulado é calculado e obtido a partir da distância entre as instâncias vizinhas e a instância de teste, e cada erro é ponderado de acordo com as distâncias euclidianas calculadas. Quanto maior a distância, menor o peso do respectivo erro, de tal modo que o regressor que obtiver o menor erro acumulado é selecionado. Neste sentido, não será selecionado, necessariamente, o modelo de regressão de melhor desempenho uniforme, visto que este regressor poderá ter erros menores para as instâncias mais distantes da região de competência. Neste método é implementado uma seleção dinâmica simples, sem utilização de *ensemble* e, consequentemente, a estimativa final é dada pelo regressor selecionado.

A segunda técnica proposta *Dynamic Weghting* (DW) combina todos os regressores do *ensemble*, usando a média ponderada. Para cada instância de teste X_j, é definida uma região de competência C, em que as instâncias pertencentes a C são parte do conjunto de validação, o qual é definido pelo valor de K. Semelhante à primeira técnica, para cada instância da região de competência é calculado um peso dado pela Equação 3.1:

$$\mathrm{D_k} = \frac{\frac{1}{d_k}}{\sum_{j=1}^{K} \frac{1}{dist_j}} \tag{3.1}$$

onde d_k é a distância entre a instância $c_k \in C$ e a instância de teste t_j.

O vetor $\vec{D_k}$ é usado para calcular o peso w_i do regressor r_i, através da equação 3.2:

$$\mathrm{w_i} = \frac{\frac{1}{\sqrt{\sum_{k=1}^{k}(D_k \times eq_{k,i})}}}{\sum_{n=1}^{N} \frac{1}{\sqrt{\sum_{k=1}^{k}(D_k \times eq_{k,n})}}} \tag{3.2}$$

onde N é o tamanho do *ensemble*, k é o índice do vizinho mais próximo e eq é o erro quadrático do regressor r_i calculado na instância de validação $c_k \in C$.

A terceira técnica, que combina um subconjunto de regressores, aborda uma seleção dinâmica de *ensembles* - *Dynamic Weighting Selection* (DWS). Os regressores com um erro acumulado acima da metade do erro intervalar $E_i = \frac{E_{max} - Emin}{2}$ são eliminados. Em que, E_{max} é o maior e E_{min} o menor erro acumulado de um regressor. A medida e a estratégia usadas para calcular o desempenho dos regressores do conjunto são as mesmas usadas na segunda técnica, conforme as equações 3.1 e 3.2.

Segundo (ROONEY et al., 2004), os métodos de seleção dinâmica avaliados tiveram desempenhos superiores aos regressores individuais, entretanto, entre as técnicas de seleção dinâmica investigadas, a segunda (DW) obteve resultados melhores do que as demais técnicas. As três técnicas também foram comparadas em (MOREIRA et al., 2009), que nesse estudo, apresentou uma estratégia diferente da abordada nos estudos anteriores, já que a comparação foi realizada através de *ensembles* heterogêneos. Os *ensembles* foram formados com quatro modelos individuais e avaliados com diversos tamanho da região de competência. Os autores concluíram que as técnicas DW e DWS foram superiores a primeira e, além disso, foi mostrado que o tamanho da região de competência é dependente do domínio do problema e, por essa razão, torna-se um fato que pode influenciar no desempenho do KNORA (KO; SABOURIN; JR., 2007).

Foi proposto em (ROONEY; PATTERSON, 2007) mais uma técnica usada em problemas de regressão, que utiliza o *Stacking* (WOLPERT, 1992) e o DWS, em que o valor da saída estimada é a média ponderada das dos dois métodos. Os pesos de cada método é ponderado na fase de treinamento e validação. Na utilização do método proposto por (ROONEY; PATTERSON, 2007), sempre dois modelos são utilizados, no entanto, as técnicas de DS e DW podem ser usadas no lugar DWS, mas essa decisão deve ser tomada na fase de validação. Os resultados do estudo mostraram que um pequeno avanço foi alcançado em relação ao desempenho do *Stacking* e da seleção dinâmica, quando usados separadamente.

Em (SERGIO; LIMA; LUDERMIR, 2016) foi proposto um método de seleção dinâmica de *ensembles*, o qual seleciona a melhor combinação de modelos de acordo com o critério usado em que, para cada instância de teste, é definido o melhor combinador na região de competência. Por exemplo, podem ser avaliados a média e a mediana das saídas de cada modelo individual para cada instância na região de competência. A abordagem desta proposta é diferente das apresentadas nos parágrafos anteriores, pois nela a função

de combinação é selecionada dinamicamente. Os experimentos mostraram que a seleção dinâmica dos combinadores foi superior à utilização de cada um estaticamente. Além disso, a técnica proposta também obteve melhor desempenho do que cada modelo individual do *ensemble* estático.

Assim como foi proposto por (SERGIO; LIMA; LUDERMIR, 2016), em (MOURA; CAVALCANTI; OLIVEIRA, 2019), foi investigada uma seleção dinâmica a partir de diferentes medidas. No total, foram comparadas oito medidas de competência usando os algoritmos DS, DW e DWS e, ao final, o *ensemble* homogêneo gerado foi composto por diversas árvores de classificação e regressão. Semelhante ao que costuma ocorrer no desempenho dos modelos individuais, onde nenhum modelo consegue superar os demais em todas as bases de dados, o estudo mostrou que nenhuma das medidas conseguiu ser superior às demais, em todas as bases de dados avaliadas, o que leva a acreditar que, para cada base de dados, uma medida ou um conjunto de medidas deveria ser usado para um domínio específico. Consequentemente, foi concluído que a seleção da melhor medida deve ser adequada à cada contexto de dados, o que a torna dependente do problema. No entanto, os autores sugerem que, ao invés de escolher a melhor medida, pode ser definido o melhor conjunto de medidas, a fim de serem obtidos melhores desempenhos nas estimativas finais.

Em (CRUZ; SABOURIN; CAVALCANTI, 2015), foram propostos algoritmos que utilizam seleção dinâmica em *ensembles* homogêneos. Os modelos homogêneos foram gerados a partir de *Random Subspace* (HO, 1998b) e o desempenho dos modelos bases foi avaliado a partir do error acumulado na região de competência. Este termo refere-se ao espaço de dados no conjunto de treinamento que mais se assemelha da instância que está sendo avaliada. Os autores do estudo concluíram que a seleção dinâmica superou os modelos bases do *ensemble*. Em complemento a este estudo, foi investigada em (MOREIRA et al., 2009) uma abordagem usando seleção dinâmica que ponderou os erros dos modelos pela distância entre o exemplo de teste e os vizinhos mais próximos na base de treinamento. Este estudo mostrou que a seleção dinâmica de múltiplos modelos heterogêneos foi superior à seleção dinâmica de um único modelo.

Em complemento, Dietterich (DIETTERICH, 2000) também argumenta que modelos múltiplos heterogêneos alcançam melhores performances quando são levadas em conta a acurácia e estabilidade dos modelos individuais. Além disso, apesar deles serem menos usados do que os homogêneos, os *ensembles* heterogêneos também alcançam resultados desejáveis (MOREIRA et al., 2012). Entretanto, as estratégias comumente usadas para construir tais modelos não são dinâmicas, o que pode levar a falhas quando combinadas (KOCAGUNELI; MENZIES; KEUNG, 2012; KITTLER et al., 1998), o que incentiva a investigação do uso de *ensembles* dinâmicos. Neste sentido, uma revisão em *Dynamic Ensemble Selection* (DES) que foi apresentada por (BRITTO; SABOURIN; OLIVEIRA, 2014), e (KO; SABOURIN; BRITTO JR., 2008), mostrou uma evolução de DS para DES. Entretanto, nenhum desses estudos investigou uma abordagem semelhante ao que é proposto nesta tese

de doutorado.

Por fim, um método proposto por (Di Nucci et al., 2017), que realiza a seleção dinâmica simples entre um conjunto de classificadores, e os métodos de seleção dinâmica que têm sido mais utilizado na literatura, o *Dynamic Classifier Selection By Local Accuracy* (DCS-LA) (WOODS; KEGELMEYER; BOWYER, 1997) e o *K-Nearest Oracle* (KNORA) (KO; SABOURIN; JR., 2007), segundo (BRITTO; SABOURIN; OLIVEIRA, 2014), compuseram a pilha de métodos de avaliação usados neste trabalho. O DCS-LA realiza uma seleção dinâmica simples, e o KNORA propõe uma seleção dinâmica de *ensembles*. Para cada um desses métodos, é possível utilizar variações mudando o valor de k, que determina o tamanho da região de competência, o que altera a estratégia de seleção de modelos que irão compor o *ensemble* dinâmico. Desta forma, cinco métodos de seleção dinâmica foram usados como *baselines* e citados no Capítulo 2. Os estudos (MOREIRA et al., 2012) e (KO; SABOURIN; BRITTO JR., 2008) apresentam uma abordagem mais ampla do uso de *ensembles* em problemas de regressão, e sobre os avanços da seleção dinâmica de múltiplos modelos, respectivamente. A seguir é apresentado um levantamento de estudos que abordam métodos de Aprendizagem de Máquina (AM) em problemas de EES e em EDM.

3.2 APRENDIZAGEM DE MÁQUINA APLICADA À EES

EES é a atividade mais comum para estimar custos e prazos em projetos de engenharia de *software*, sendo o número de pessoas-horas, dias ou meses usado para estimar cada tarefa do projeto. O esforço representa o maior fator de custo do software (ARGAWAL et al., 2001), no entanto, o principal problema para prever o esforço de trabalho em um projeto de *software* é o estabelecimento de um conjunto de dados históricos composto de atributos e valores, assim como quantificar os atributos em termos numéricos (ARAUJO et al., 2012). Neste sentido, normalmente a variável dependente é representada pelo o esforço dado em horas (ARAUJO; SOARES; OLIVEIRA, 2012), porém, alguns bancos de dados históricos também consideram a predição do esforço em tempo de entrega (OLIVEIRA et al., 2010).

Buscando atender às demandas correntes em processos de EES, o gerente de projetos precisa estimar a duração e o esforço correspondente (OLIVEIRA, 2006). Portanto, o principal fator de risco para o desenvolvimento de *software* é o custo para concluí-lo. Superestimativas podem levar o cliente a desistir do projeto, enquanto estimativas muito otimistas tendem a aumentar consideravelmente os recursos necessários para conclusão do projeto dentro do prazo subestimado. Diante disso, as particularidades no desenvolvimento de um projeto de *software* pode tornar o processo de estimativa muito difícil (ARAUJO; SOARES; OLIVEIRA, 2012) e, por esta razão, EES é ainda considerado um desafio dentro da engenharia de *software*.

Desde a década de 1980, vários modelos têm sido propostos na literatura de EES. Essas técnicas podem ser agrupadas em várias categorias, por exemplo: (i) experiência de especialistas, que consiste na consulta de especialistas para prever o esforço do *software*;

(ii) técnicas paramétricas, que são derivadas da estatística e de dados históricos; e (iii) técnicas de inteligência artificial que são baseadas em métodos de aprendizagem de máquina. Porém, alguns autores definem outros grupos de técnicas de estimativas, algumas vezes mais abrangentes ou apresentadas em uma perspectiva diferente, como é o caso, por exemplo, grupo de métodos não algorítmicos e algorítmicos, além de subdivisões desses grupos.

Apesar das diversas técnicas de EES, métodos baseados em AM estão sendo cada vez mais usados em problemas deste tipo, e o uso dessas técnicas vem crescendo nos últimos anos (WEN et al., 2012). Além disso, vários pesquisadores já consideram as técnicas de AM como uma categoria de técnicas de EES (BOEHM; SULLIVAN, 1999; MENDES et al., 2003). Na revisão sistemática (WEN et al., 2012), os autores concluíram, de maneira geral, que os modelos baseados em aprendizagem de máquina são mais acurados do que modelos que não utilizam aprendizagem. Nesse sentido, consideramos que os métodos de AM também podem ser vistos como técnicas de EES. Neste trabalho, modelos baseados em: (i) analogia, (ii) regressão, (iii) probabilísticos, (iv) redes neurais e (v) árvores foram classificados como algoritmos de AM e avaliados nos experimentos dentro do grupo de métodos de avaliação.

A AM tem sido usada para resolver diferentes tipos de problemas e, desde a década de 80, muitos artigos têm sugerido métodos de AM para melhorar o desempenho em EES. Segundo a revisão sistemática de métodos de AM em EES realizada em (WEN et al., 2012), as técnicas com maior frequência de uso são: *Case-Based Reasoning* (CBR), *Artificial Neural Network* (ANN), *Decision Trees* (DT), *Bayesian Network* (BN) e *Support Vector Regression* (SVR). Entretanto, muitos estudos mostram que métodos de similaridade tendem a ser uma boa opção (SHEPPERD; SCHOFIELD, 1997; WALKERDEN; JEFFERY, 1999), enquanto outros mostram que os métodos de AM têm superado os de regressão linear (GRAY; MACDONELL, 1997), e ainda que métodos de regressão superam alguns métodos de AM (MENDES et al., 2003; BRIAND et al., 1999; STENSRUD, 2001). De fato, não existe uma técnica que seja consensualmente superior.

No entanto, existem técnicas que são comumente usadas e que alcançam bons resultados. Por exemplo, o SVR é um modelo amplamente usado no campo da Engenharia de *Software* (ES), especialmente em problemas com alta dimensionalidade e tendências a conter ruídos. O SVR tem mostrado resultados relevantes, em base de dados com essas características (CORAZZA et al., 2011; OLIVEIRA, 2006).

Em (DRAGICEVIC; CELAR; TURIC, 2017), os autores propuseram um modelo de BN capaz de prever o esforço em qualquer método de desenvolvimento ágil. Trata-se de um pequeno e simples método que não impacta nas práticas ágeis. Uma investigação comparativa entre *Multi Layer Perceptron* (MLP) e *Radial Basis Function Neural Network* (RBFNN), contra *Multi Linear Regression* (MLR), foi realizada em (LóPEZ-MARTíN; ABRAN, 2015), e os resultados demonstraram que os modelos de MLP e RBFNN superaram os de MLR. Entretanto, os desempenhos dos modelos são alterados, quando os bancos de dados mu-

dam, uma vez que os dados e os hiper parâmetros de otimização dos algoritmos podem interferir significativamente nos resultados.

Diante dessas pesquisas, e do grande número de trabalhos conhecidos na área de EES (WEN et al., 2012), é possível dizer que modelos de AM já estão consolidados na comunidade de ES, e que certamente não existe um modelo matemático para resolver todos os problemas da área, mesmo se fosse considerado apenas problemas de estimativa de esforço de *software*, e no entanto, a combinação desses modelos tem crescido dentro deste contexto. Os sistemas de múltiplos modelos tentam melhorar o desempenho previamente alcançado pelos modelos bases ou individuais.

Neste sentido, a literatura de métodos de EES define *Ensemble Effort Estimation* (EEE) como uma combinação de vários modelos individuais, ou modelos bases, combinados através de uma regra específica (SENI; ELDER, 2010) para EES. Trabalhos anteriores investigaram o uso de *ensembles* estáticos e os resultados têm mostrado que EEE geralmente são mais eficientes (TWALA; VERNER, 2016; IDRI; HOSNI; ABRAN, 2016b; IDRI; HOSNI; ABRAN, 2016a). Esses estudos mostraram que a combinação de múltiplos modelos de AM melhoram a precisão da EES. Braga et al. (BRAGA et al., 2007) afirmou que *Bagging* melhora a estimativa produzida por modelos bases, tais como *Regression Trees* (RT) e MLP. Kultur et al. (KULTUR; TURHAN; BENER, 2009) utilizaram cinco bases de dados e mostraram que uma versão adaptada de *Bagging* provê grandes melhorias em comparação aos modelos bases. O artigo (HOSNI et al., 2016) propôs e avaliou *ensembles* heterogêneos baseados em quatro modelos de AM bem conhecidos (KNN, MLP, SVR e *M5 Base* (M5P)), usando três regras de combinação linear em dois conjuntos de dados. Os resultados empíricos do estudo mostraram que o *ensemble* proposto melhorou o desempenho das estimativas dos métodos individuais em um banco de dados (Myazaki), e alcançou a segunda melhor performance no outro (Albrecht).

Um mapeamento sistemático considerando estudos publicados entre 2000 e 2015 resumiu os trabalhos existentes em EEE e foi apresentado em (IDRI; HOSNI; ABRAN, 2016b). Somado a esse, uma revisão sistemática da literatura em EEE também foi apresentada em (IDRI; HOSNI; ABRAN, 2016a). Os autores concluíram que *ensembles* melhoram os resultados alcançados por modelos individuais e que essa estratégia vem sendo usada na literatura de EES, mas ainda não tem uma representação massiva. As principais descobertas deste último estudo foram: (i) EEE não tem sido massivamente investigado; (ii) *ensembles* homogêneos foram os tipos mais investigados em EEE; (iii) basicamente dois tipos de combinações foram usadas para prever o esforço em técnicas de EEE, linear e não linear; (iv) modelos de AM foram as técnicas individuais mais investigadas para construir *ensembles* heterogêneos em EES, e (v) *ensembles* heterogêneos são mais acurados do que os métodos bases.

Complementando a revisão dos estudos, foi realizada uma busca na literatura de EEE entre os anos de 2016 e 2020 e os resultados alcançados foram semelhantes aos apresen-

tados por (IDRI; HOSNI; ABRAN, 2016a). Os estudos selecionados não apresentaram DES em suas estruturas, exceto (CABRAL et al., 2017) que tem relação direta com este trabalho. Neste sentido, foi desenvolvido em (CABRAL; OLIVEIRA; da Silva, 2022) um estudo atualizado da revisão (IDRI; HOSNI; ABRAN, 2016a). A seguir é apresentada as principais descobertas do novo estudo.

- A proporção de estudos que utilizaram ensembles homogêneos diminuiu em relação aos ensembles heterogêneos de 2016 a 2020 (de 70% para 60%). No geral, 17 estudos usaram *ensembles* homogêneos, enquanto 16 investigaram os heterogêneos. Além disso, devemos destacar que alguns pesquisadores discutiram os dois tipos de ensembles no mesmo estudo.
- As ANN foram as técnicas mais utilizadas em *ensembles* homogêneos e heterogêneos, com ênfase em nas redes MLP.
- A média foi a regra de combinação adotada na maioria dos estudos, seguida da mediana, que obteve melhores resultados do que a média em um contexto geral.
- Cocomo81 e Desharnais foram as bases de dados mais utilizadas nas pesquisas de EEE; no entanto, o uso das bases de dados Albrecht, ISBSG e Kemerer aumentou nos últimos anos. Tukutuku, China e alguns outros bancos de dados proprietários e também abertos não foram encontrados nos estudos selecionados no trabalho anterior (IDRI; HOSNI; ABRAN, 2016a), mas tem crescido nos últimos anos e tendem a continuar sendo utilizados. O uso do Cocomonasa e SDR diminuíram.
- As métricas *Mean Magnitude Relative Error* (MMRE), *Percentage Relative Error Deviation* (PRED) e *Median Magnitude Relative Error* (MdMRE) ainda são amplamente utilizadas, no entanto, houve um aumento evidente no uso da *Mean Absolute Error* (MAE) e do *Standard Accuracy* (SA). Com relação ao MAE, a métrica vem sendo amplamente utilizada nos últimos anos, e o SA teve um crescimento percentual. Essas mudanças no uso das métricas foram influenciadas pelo viés encontrado no MMRE (SHEPPERD; MACDONELL, 2012).
- Leave-one-out foi o método de validação mais utilizado. Isso provavelmente ocorreu devido ao tamanho relativamente pequeno dos bancos de dados.
- Os testes estatísticos de *Wilcoxon* e de *Friedman* são os mais comuns nos estudos que apresentaram os resultados por meio de experimentos através de testes de hipóteses. A análise dos dados tende a não seguir uma distribuição normal, o que leva as comparações a serem feitas com testes não paramétricos em pesquisas de EEE.
- EEE melhorou a precisão dos métodos individuais, pois, mais de 95% dos estudos relevantes que o trabalho reuniu apresentou melhorias no uso de *ensembles* dentro de EES, sendo que apenas um estudo não apresentou melhorias.

- A mediana e o *Inversed Ranking Weighted Mean* (IRWM) levaram uma pequena vantagem em relação aos tipos de métricas de combinação, mas não podemos afirmar que essas métricas superam as demais, quanto à maneira de combinar métodos individuais em EEE, pois não existe uma regra de combinação que seja significativamente melhor que as demais.

- A seleção dinâmica ainda é pouco utilizada em EEE, no entanto, já existe estudo que mostra possíveis melhorias em comparação com os *ensembles* estáticos.

Resumindo as principais revisões da literatura nas áreas abordadas neste trabalho, tem-se um mapeamento sistemático realizado entre 2000 e 2015 que sumarizou estudos em EEE (IDRI; HOSNI; ABRAN, 2016b) e, uma revisão sistemática da literatura de EEE (IDRI; HOSNI; ABRAN, 2016a), que resumiu e reuniu os resultados empíricos de diversos estudos publicados entre 2000 e 2015. Nesses estudos, os autores concluíram que a construção de EEE geralmente melhora os resultados alcançados por métodos individuais, conforme os que foram apresentados na revisão sistemática (WEN et al., 2012).

A literatura também não apresentou estudos de EES que envolvessem a combinação e seleção dinâmica de múltiplos modelos de regressão heterogêneos. Essa proposta consiste em buscar melhorarias nas acurácias alcançadas em problemas de EES, o que se resume basicamente a modelos de regressão selecionados dinamicamente por classificadores. Diante das vantagens apresentadas nos estudos anteriores quanto ao uso de: (i) múltiplos modelos; e (ii) seleção dinâmica, podemos imaginar que é possível avançar na precisão das estimativas em problemas de EES, inicialmente e, posteriormente, em diferentes contextos.

Nesse sentido, considerando que: (i) a maioria dos bancos de dados relacionados à EES são pequenos, quando comparados com outros conjuntos de dados; (ii) os resultados mais acurados que um sistema de múltiplos modelos pode alcançar depende da qualidade dos dados e da seleção apropriada dos algoritmos; (iii) recentes trabalhos de seleção dinâmica têm demonstrado que técnicas de DS são ferramentas eficientes para problemas mal definidos (CAVALIN; SABOURIN; SUEN, 2013); (iv) nas base de treinamento não existem dados suficientes para construir modelos individuais consistentes (CAVALIN; SABOURIN; SUEN, 2013); e finalmente (v) que o uso de DES tem superado os *ensembles* estáticos e seleção dinâmica de um único modelo (KO; SABOURIN; BRITTO JR., 2008); este trabalho propõe um *framework* de DES composto por um conjunto de modelos de regressão heterogêneos selecionados dinamicamente por classificadores, sendo inicialmente aplicado a projetos de EES e de EDM.

3.3 APRENDIZAGEM DE MÁQUINA APLICADA À EDUCAÇÃO

O uso de algoritmos de AM para a exploração de dados educacionais é um campo emergente destinado a desenvolver métodos de estimativas importantes no contexto educacional, assim como para a descoberta de padrões significativos (KOTSIANTIS, 2012). Neste

sentido, dados escolares são usados como entrada dos algoritmos que criam modelos capazes de prever valores que dão suporte à administração escolar. Por exemplo, atributos como as principais características demográficas dos alunos e as suas notas em testes de avaliação poderão constituir um conjunto de dados de treinamento para um algoritmo de regressão. Ou seja, os algoritmos, juntamente com os dados de entrada, são utilizados para gerar modelos de AM e, consequentemente, prever o desempenho dos alunos e de outras variáveis do contexto educacional.

Diversas aplicações ou tarefas em ambientes educacionais podem ser realizadas através de algoritmos de AM. É sugerido em (BAKER; YACEF, 2009; BAKER; MCGAW; PETERSON, 2010) quatro áreas chaves que são frequentemente usadas para a aplicação de algoritmos de AM. Essas áreas constituem estudos para melhorar: (i) o padrão dos alunos; (ii) o padrão de dados do domínio; (iii) os estudos que dão suporte pedagógico à equipe de educadores através de *softwares* de aprendizagem e (iv) as pesquisas científicas relacionadas à aprendizagem dos alunos. Além dessas áreas de estudo, cinco abordagens de AM são costumeiramente realizadas: (i) a previsão das variáveis de interesse; (ii) o agrupamento de dados similares; (iii) a análise de relacionamento entre variáveis; (iv) o detalhamento descritivo de dados para o julgamento humano e (v) a descoberta de padrões. Os experimentos realizados neste trabalho abordaram a previsão da aprendizagem dos alunos de ensino fundamental e médio. Não obstante, é proposto por (CASTRO et al., 2007) aplicações que lidam com a avaliação do desempenho dos alunos; aplicativos que fornecem adaptação e recomendação de cursos; abordagens que tratam da avaliação de material de aprendizagem; aplicativos que envolvem *feedback* para professores e alunos quanto aos cursos de aprendizado a distância; e o desenvolvimento de aplicações para detectar comportamentos de aprendizado atípicos nos estudantes.

No entanto, ainda é possível que sejam adicionadas outras categorias de estudo ao grupo de tarefas educacionais que aplicam AM. Estas categorias podem vir de diferentes unidades de pesquisas e utilizar vários algoritmos de AM individualmente ou em conjunto. Este trabalho também abordou experimentos sobre a previsão de desempenho dos alunos, que é uma categoria ou linha de pesquisa com diversos estudos relacionados. Segundo a revisão realizada em (ROMERO; VENTURA, 2010a), a previsão de desempenho dos alunos é a segunda linha de pesquisa mais avaliada, sendo superada apenas pela área de desenvolvimento de aplicações que fornecem um *feedback* para alunos e professores. Quanto às tarefas de AM, as mais comumente utilizadas em ordem de uso são: regressão, agrupamento, classificação e associação; e dentre os algoritmos mais usados, tem-se: as árvores de decisão, as redes neurais e os algoritmos baseados em probabilidade (ROMERO; VENTURA, 2010a). Todas as avaliações de desempenho analisadas neste trabalho foram realizadas através das previsões das taxas de evasão, aprovação e reprovação dos alunos por escolas.

O objetivo da predição de desempenho é estimar o valor não conhecido de uma variável

que descreve o aluno ou estima uma taxa de avaliação escolar. Os valores de desempenho, conhecimento, pontuação ou nota são normalmente previstos com o apoio de algoritmos de AM. Consequentemente, o uso destes algoritmos na previsão de desempenho dos alunos é uma das áreas mais antigas e populares em EDM, sendo que, boa parte dos estudos relacionados à aplicação de algoritmos de AM é relacionada com a avaliação de desempenho dos alunos. Por exemplo, em (HäMäLäINEN; VINNI, 2006), foi realizada uma comparação entre métodos de AM para prever a probabilidade de aprovação e reprovação dos alunos, enquanto no estudo (ROMERO et al., 2008) foi apresentada uma comparação entre diferentes algoritmos de AM para nivelar os alunos, com base nos dados do *Moodle* (uma ferramenta de educação a distância). Neste, a previsão dada se referiu às notas finais dos alunos. Mais dois estudos de previsão de desempenho foram abordados em (MINAEI-BIDGOLI et al., 2003; FENG, 2019); o primeiro estimou o desempenho dos alunos com base em dados históricos da escola, enquanto o segundo fez previsões do desempenho de alunos universitários, utilizando modelos de árvores de decisão e redes neurais. Porém, as análises realizadas nos estudos não se resumem à avaliação de desempenho dos alunos, pois, outras linhas de pesquisas, como a previsão de acertos em questões de prova, também foram abordadas em diversos estudos relacionados à aplicação de AM na educação, inclusive, o estudo (BECK; WOOLF, 2000), que inspirou outros a desenvolverem modelos de previsão acertos em questões, foi um dos primeiros publicados na área. A seguir será apresentado o uso de alguns algoritmos de AM no contexto educacional.

As redes neurais artificiais têm sido usadas de diferentes maneiras para prever as notas finais dos alunos há décadas (GEDEON; TURNER, 1993). Elas utilizam desde as arquiteturas de *backpropagation* as de *feed-foward*. Em (WANG; MITROVIC, 2002), elas foram usadas para prever o número de erros que os alunos iriam cometer em um problema e, em seguida, sugerem um exercício adequado para o aluno de acordo com desempenho dele nos anteriores; em (FAUSETT; ELWASIF, 1994), duas redes neurais são usadas antever o desempenho dos alunos na disciplina de Cálculo, a partir das respostas do teste de nivelamento. O *Moodle*, conforme citado no parágrafo anterior, é uma ferramenta que prover uma fonte de dados educacionais que podem ser usados para as tarefas de aprendizagem. Em (DELAVARI; PHON-AMNUAISUK; ZADEH, 2008), a ferramenta foi novamente usada para fornecer dados a um algoritmo de rede neural com função *base radial* para estimar as notas dos alunos. O mesmo estudo também propôs diretrizes para as instituições de ensino superior, diretrizes essas que buscam aprimorar os processos de decisões atuais da escola através da aplicação de algoritmos de AM, os quais consequentemente, tentam descobrir novos conhecimentos úteis para os processos de tomadas de decisão. Por fim, uma MLP foi usada para prever o desempenho de um candidato admitido a uma vaga em uma universidade (LIVIERIS; DRAKOPOULOU; PINTELAS, 2012), a partir do desenvolvimento de uma ferramenta com interface simples que pode ser utilizada por um educador para classificar os alunos e distinguir aqueles com maior probabilidade de baixo aprovei-

tamento e, em (WANG; LIAO, 2011), outra rede neural foi usada para adaptar exercícios conforme o aprendizado dos alunos em um curso de língua inglesa, criando experiências de aprendizagem ideais para cada aluno. Conforme apresentado, as redes neurais são um dos algoritmos mais frequentes quanto à aplicação de AM na educação.

Os algoritmos baseados em probabilidades, em especial as *Bayesian Networks*, também já foram usados no domínio de dados educacionais (HIEN; HADDAWY, 2007), os autores modelaram um algoritmo de AM para prever as médias acumuladas dos alunos de graduação, com base no histórico do candidato no momento da admissão. O uso deste tipo de algoritmo varia, desde a avaliação do desempenho do aluno dentro de um sistema de tutoria inteligente (PARDOS et al., 2007) até a criação de modelos para determinar a probabilidade de uma questão de múltipla escolha ser marcada corretamente (PARDOS et al., 2008; DESMARAIS; GAGNON, 2006). Outras áreas, como a aprendizagem colaborativa (STEVENS et al., 2005) e o desempenho dos alunos quanto ao exame final com tutores *on-line* (AYERS; JUNKER, 2006) também foram avaliados.

Sistemas baseados em regras são simples de compreender devido à estrutura utilizada pelos modelos criados. O desempenho dos alunos foi previsto em um ambiente de educação a distância usando regras de associação *fuzzy* em (NEBOT; ESPINOZA; VELLIDO, 2006); no mesmo estudo, também foram verificadas as características relevantes dos dados que foram envolvidos no processo de avaliação, enquanto outro sistema foi baseado em portfólios de aprendizagem, usando regras formativas (CHEN; CHEN; LI, 2007). Ainda, regras de indução foram usadas para prever o desempenho acadêmico dos alunos e os resultados foram apresentados em (OGOR, 2007). Uma abordagem diferente de aprendizado de regras foi aplicada no estudo (SHANGPING; PING, 2008), em que um novo algoritmo genético foi usado para encontrar regras de associação, com o intuito de prever notas finais dos alunos, com base em dados gravados em um sistema educacional na *Web*. Por fim, mais um modelo de extração de regras foi usado para estimar notas dos alunos em um ambiente *on-line* (ETCHELLS et al., 2006).

Árvores de regressão, redes neurais, regressão linear e máquinas de vetor de suporte também são algoritmos que já foram usados para gerar modelos de regressão para prever notas de alunos. Em (KOTSIANTIS; PINTELAS, 2005), a previsão de desempenho dos alunos foi aplicada uma universidade de dados educacionais abertos. A previsão de prestação de contas dos alunos no final do ano também é uma linha de pesquisa investigada com algoritmos de AM (ANOZIE; JUNKER, 2006). Uma regressão multivariada foi usada em (YAVUZ, 2008) para prever o desempenho dos alunos, e outra *step by step* foi apresentada em (HELLAS et al., 2018). Outra área de pesquisa em dados educacionais é a avaliação de materiais. Em (ARNOLD et al., 2005), foi desenvolvida uma MLR para prever o tempo gasto por um aluno para o aprendizado de uma página de material, no entanto, os estudos que aplicam AM em dados educacionais não são voltados apenas para predição. Em (MARTÍNEZ, 2001), foi usada uma MLR para identificar as melhores variáveis preditoras

para uma variável dependente. A análise de satisfação do estudante (THOMAS; GALAMBOS, 2004) e os resultados de cursos de educação a distância (MYLLER; SUHONEN; SUTINEN, 2002) foram problemas analisados com o uso de regressão linear e árvores de decisão, respectivamente. Uma regressão *Ridge* avaliou em (CETINTAS et al., 2009) a probabilidade de um aluno acertar uma questão. Por fim, a previsão do desempenho dos alunos em estudos *on-line* foram analisadas através de análises de correlação em (WANG; NEWLIN, 2002; PRITCHARD; WARNAKULASOORIYA, 2005).

Estudos mais recentes do que os mencionados nos parágrafos anteriores relatam a contínua utilização de modelos de AM em dados educacionais, conforme pode ser verificado em (SPIKOL; RUFFALDI; CUKUROVA, 2017; SANTOS; RODRíGUEZ; PINTO-LLORENTE, 2020; NEBOT; ESPINOZA, 2020; INUSAH, 2022; RANGONE; PIZARRO; MONTEJANO, 2022; NAFIE; HAMED, 2021). Além desses, revisões mais recentes da literatura foram realizadas em (SHAHIRI; HUSAIN; RASHID, 2015; UKWUOMA et al., 2019; LYNN; EMANUEL, 2021). Mesmo os modelos individuais de regressão podendo ser usados para resolver os problemas mencionados, o poder preditivo dos modelos construídos pode ser melhorado através de *ensembles* (NASCIMENTO; FAGUNDES; MACIEL, 2019). Os modelos baseados em *ensembles* geralmente resultam em melhor desempenho e são mais estáveis do que os modelos individuais, porque combinam a previsão de seus componentes, proporcionando um resultado mais robusto (GARCIA-PEDRAJAS; HERVAS-MARTINEZ; ORTIZ-BOYER, 2005; KOTSIANTIS; PATRIARCHEAS; XENOS, 2010; BEEMER et al., 2017b; NASCIMENTO; FAGUNDES; MACIEL, 2019).

Em relação à abordagem de *ensemble* aplicada ao cenário educacional, em (KOTSIANTIS; PATRIARCHEAS; XENOS, 2010) um *ensemble* de classificadores é proposto como um direcionamento para melhorar a capacidade preditiva do desempenho de um aluno associado ao ensino universitário a distância. Em (BEEMER et al., 2017b), é apresentada uma abordagem de *ensembles* para executar tarefas analíticas com foco específico na estimativa de efeitos de tratamento individualizados, os quais funcionam como medidas que permitem, para cada aluno, quantificar o impacto da estratégia de intervenção no desempenho do aluno. Por fim, em (NASCIMENTO; FAGUNDES; MACIEL, 2019), foram previstos indicadores educacionais relacionados aos índices de eficiência escolar no cenário brasileiro, e nesse estudo foi comparado o desempenho de dois *ensembles*, utilizando modelos individuais heterogêneos e de regressão. O *Stacking* foi usado para construir os ensembles avaliados em (NASCIMENTO; FAGUNDES; MACIEL, 2019), cujos dados usados no estudo também foram avaliados nos experimentos apresentados no Capítulo 6 deste trabalho.

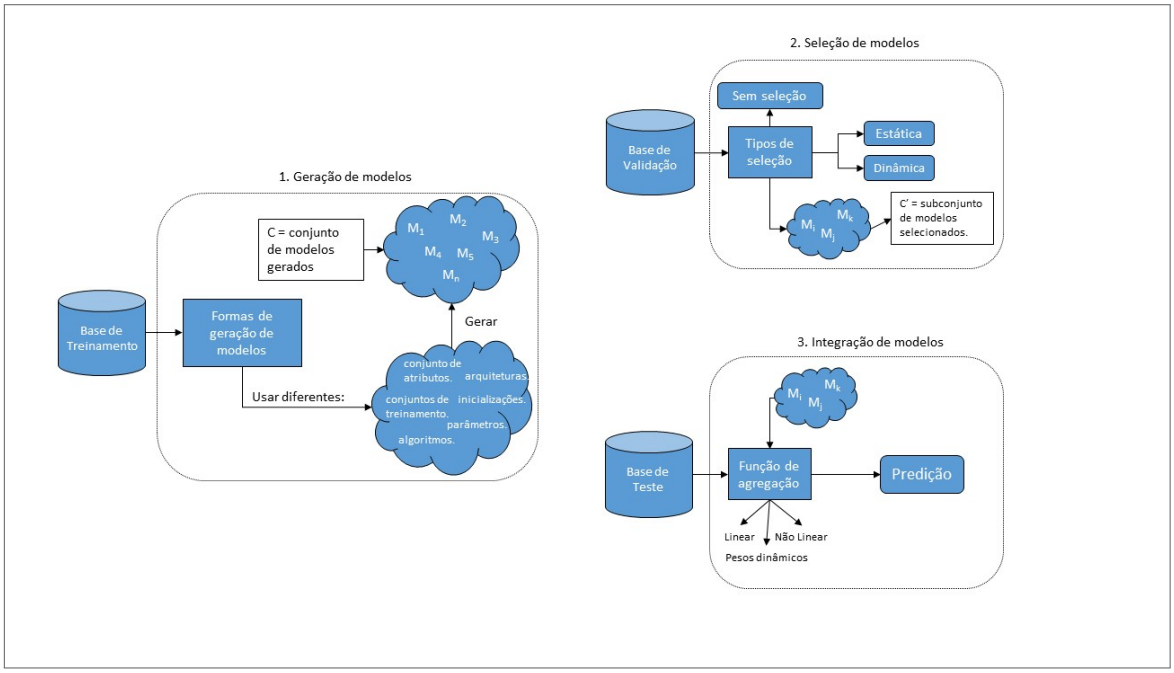
4 FRAMEWORK PARA SELEÇÃO DINÂMICA DE MÚLTIPLOS MODELOS

Neste capítulo, é apresentada as etapas do *framework* proposto para seleção dinâmica de múltiplos modelos heterogêneos. Segundo (CRUZ; SABOURIN; CAVALCANTI, 2018), um sistema de múltiplos modelos pode ser dividido em três etapas, descritas a seguir.

Na primeira, um conjunto de modelos $M = M_1, M_2, M_3, M_4, M_5, ..., M_n$ (sendo n a quantidade de modelos) é gerado a partir de um ou vários algoritmos. Na segunda, por vezes chamada de etapa de corte, um conjunto de modelos é eliminado, e nela a seleção do subconjunto de modelos pode ser estática ou dinâmica ou até mesmo ser ignorada, o que levará todos os modelos gerados para a etapa posterior. Finalmente, na etapa de integração dos modelos, uma estratégia de combinação é definida e usada para obter a predição de novos casos que é baseada na predição dos modelos bases. Essas predições são agregadas para dar a saída final, sendo que esta agregação pode ser definida por funções lineares ou não-lineares com pesos fixos ou dinâmicos para cada modelo selecionado.

Quanto à configuração experimental dos dados, a abordagem mais comum é dividi-los em 3 partes: (i) o conjunto de treinamento, usado para obter os modelos bases ou individuais; (ii) o conjunto de validação, usado para avaliar a generalização de cada modelo e definir os hiper parâmetros dos algoritmos; e (iii) o conjunto de testes, que é usado para avaliar a performance final dos métodos investigados. A Figura 11 mostra todo o processo de construção de um Sistema de Múltiplos Modelos (SMM) com detalhes para as formas de geração de modelos e os tipos de seleção.

Figura 11 – As etapas de construção de um sistema de múltiplos modelos



Fonte: O autor (2022)

4.1 DESAFIOS

Estudos da literatura mostram que a combinação de modelos tende a aumentar a acurácia das predições (KITTLER et al., 1998; SENI; ELDER, 2010; DIETTERICH, 2000; IDRI; HOSNI; ABRAN, 2016a; IDRI; HOSNI; ABRAN, 2016b). A principal característica que existe em um SMM é a capacidade de contar com a expertise individual dos modelos, assim como, com a possibilidade deles se diferenciarem em termos de propriedades e conceitos (BRITTO; SABOURIN; OLIVEIRA, 2014). Definir modelos acurados e diversos é a chave para construir SMM de qualidade (BROWN; WYATT; TINO, 2005).

Modelos com boa performance podem ser obtidos através da alteração dos valores dos hiper parâmetros e das avaliações prévias no conjunto de validação. Enquanto a diversidade entre os modelos gerados pode ser alcançada através de diferentes estratégias, entre elas, o uso de métodos de amostragem de dados (*bootstrapping*) (BREIMAN, 1996) para *ensembles* homogêneos, ou a partir de dois ou mais algoritmos distintos, que tenham diferentes estratégias de aprendizado, para os heterogêneos. A Figura 11 ilustra diversas maneiras nas quais um SMM pode alcançar diversidade. Os *ensembles* homogêneos geralmente utilizam a estratégia de alterar o conjunto de amostras ou os hiper parâmetros dos algoritmos, a fim de alcançar diversidade, enquanto que os *ensembles* heterogêneos possuem em sua essência a característica de ser um SMM que já prover diversidade, visto que eles são compostos por modelos gerados a partir de algoritmos que contêm naturezas distintas (WEBB; ZHENG, 2004). Assim, podemos dizer que a acurácia e a diversidade são cruciais para obter melhorias através de um SMM (BROWN; WYATT; TINO, 2005; BROWN et al., 2005).

No entanto, para construir um SMM é possível ainda destacar outros dois desafios: (i) a escolha adequada dos modelos bases ou individuais que irão construir o *ensemble*, e (ii) a escolha do critério adequado para a fusão ou seleção dos modelos bases (SHUNMUGAPRIYA; KANMANI, 2013). Nos dois próximos parágrafos, abordaremos os respectivos temas.

Para solucionar o primeiro desafio, é preciso gerar diversos modelos para o domínio avaliado, alterando os hiper parâmetros deles, a fim de avaliar o desempenho individual. Os estudos de (WEN et al., 2012; IDRI; HOSNI; ABRAN, 2016a) identificaram os principais algoritmos usados em problemas de Estimativa de Esforço de *Software* (EES), enquanto algoritmos como *Naive Bayes*, *Bayes Net*, *Support Vector Machine*, *Logistic Regression* e *Decision Trees* têm sido usados na área educacional (ROMERO; VENTURA, 2010b). Todavia, ao invés de simplesmente escolher quaisquer algoritmos usados em trabalhos anteriores, realizar uma avaliação prévia de diversos modelos para um domínio específico é uma estratégia que tende a alcançar bons resultados (IDRI; HOSNI; ABRAN, 2016b). No Item 4.4.1, apresentaremos os algoritmos usados para gerar os modelos de regressão avaliados neste trabalho.

O segundo desafio apontado se refere ao critério usado para medir o nível de competência de cada modelo pertencente SMM. O uso de diferentes estratégias para integrar os

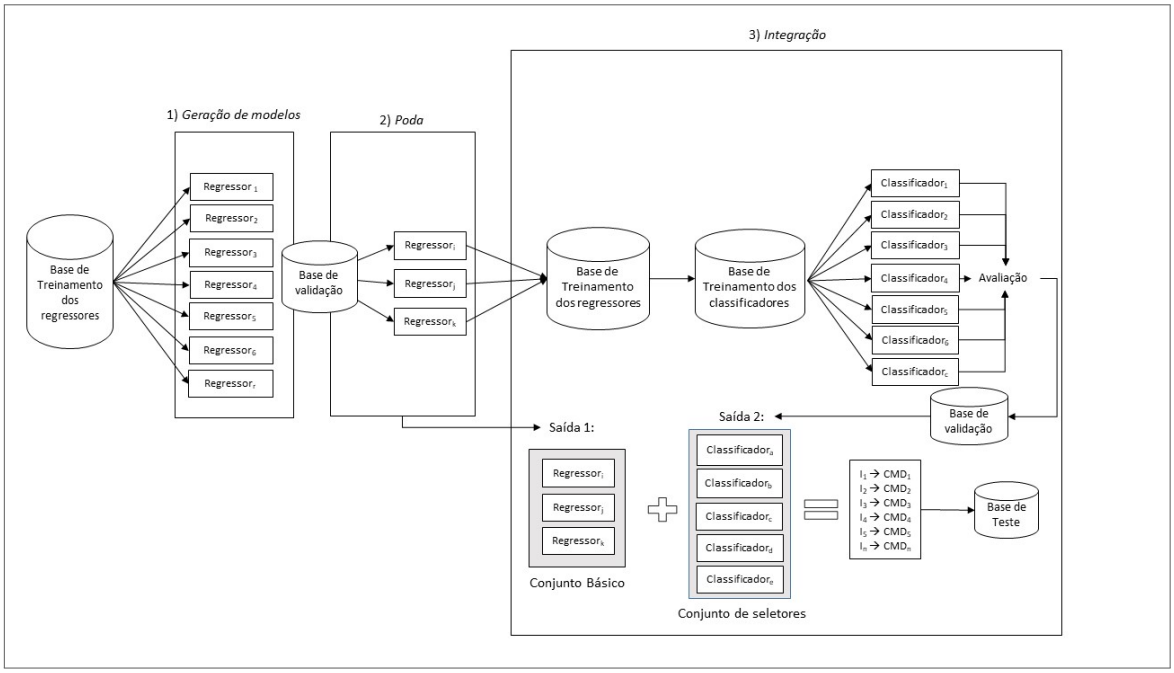
modelos pode levar o SMM a obter melhores resultados (CRUZ; SABOURIN; CAVALCANTI, 2014). Na proposta apresentada neste trabalho, classificadores são usados para realizar a seleção dinâmica de regressores heterogêneos pré-selecionados através de um processo de validação. Para simplificar e especificar a função desses modelos de classificação, os chamaremos também de seletores. Esses seletores utilizam diferentes critérios, como, por exemplo, desde a possibilidade de variar da avaliação pelo desempenho local dos modelos, até o uso de abordagens mais complexas (e.g. Redes Neurais). No Item 4.4.1, apresentaremos os classificadores usados na proposta apresentada neste trabalho.

Em resumo, os regressores investigados nesta pesquisa são usados para prever o valor da variável dependente de cada exemplo de teste, enquanto os classificadores irão selecionar os regressores adequados para constituir o SMM dinâmico. Vale destacar também que a seleção dinâmica proposta pode ser simples, quando sempre um modelo do conjunto é selecionado, ou múltipla, quando vários modelos são selecionados. O *framework* proposto constrói um SMM dinâmico, que poderá ser simples (um modelo) ou múltiplo. A seguir, serão definidas as características do *framework* que levarão a seleção de um ou vários modelos. A diversificação da seleção dinâmica será dada através de diferentes critérios, os quais irão variar de acordo com a variação dos classificadores (seletores).

4.2 ETAPAS

A Figura 12 ilustra de maneira resumida o *framework* proposto. Foi destacado nela que o *framework* envolve 3 etapas: (i) a geração; (ii) a poda; e (iii) a integração.

Figura 12 – Etapa de geração dos modelos de regressão do *Framework* proposto

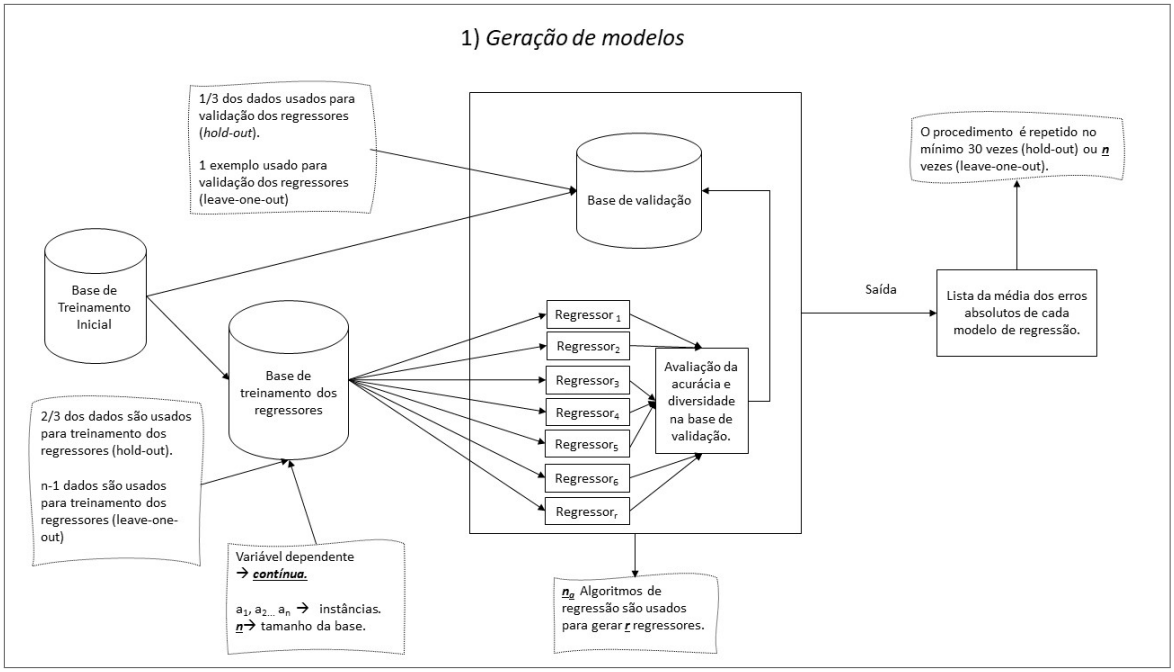


Fonte: O autor (2022)

Na primeira etapa, são definidos os algoritmos e os regressores parametrizados com diferentes valores de hiper parâmetros e que foram criados a partir desses algoritmos. A variação de parâmetros tem o intuito de encontrar modelos com melhores desempenhos, porém ela é opcional durante este processo, visto que a opção por gerar e avaliar vários modelos poderá ser custosa. A quantidade exata de algoritmos e de regressores gerados a partir de cada algoritmo não é uma constante definida no *framework*, no entanto, considerando que são utilizados no mínimo três regressores no Conjunto Básico (CB), é sugerido que sejam usados ao menos três algoritmos distintos, sendo possível, dessa forma, gerar maior diversidade entre os modelos. Ainda, a escolha do CB não é uma tarefa trivial e, por esta razão, testes estatísticos devem ser usados para minimizar o risco de viés.

Ao fim da primeira etapa, teremos um número r de regressores gerados a partir dos n_a algoritmos usados, de tal maneira que, sendo n_a o número de algoritmo e, n_r o de regressores por algoritmo, teremos $r = n_a * n_r$. Porém, se a quantidade de regressores gerados por cada algoritmo não for igual para todos os algoritmos, r será definido como a soma da quantidade de n_r de cada algoritmo, $r = n_{r1} + n_{r2} + ... + n_{rna}$. É importante destacar que na primeira etapa, o conjunto de dados de treinamento é usado pelos algoritmos de regressão com o objetivo de induzir um ou vários regressores. Na etapa seguinte, esses modelos de regressão são avaliados no conjunto de validação Base de Validação (τ_v), que é disjunto do conjunto de treinamento dos regressores Base de Treinamento dos Regressores (τ_r). A Figura 13 detalha a etapa de geração dos modelos.

Figura 13 – Etapa de geração dos modelos de regressão do *Framework* proposto



Fonte: O autor (2022)

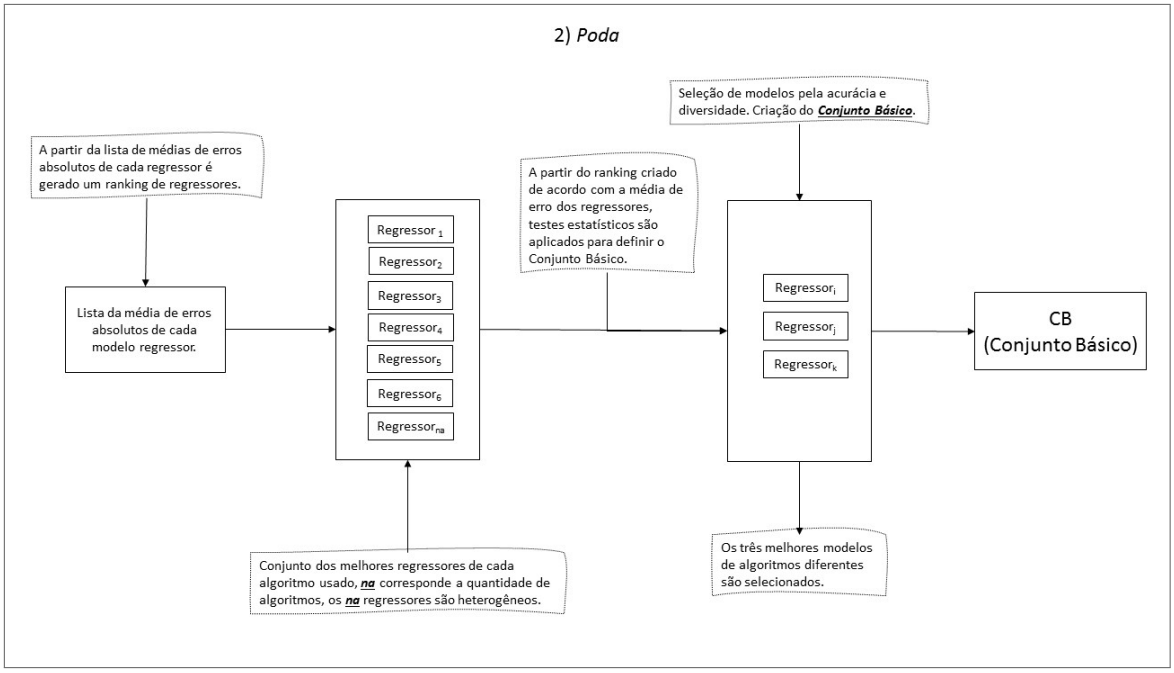
Na segunda etapa, cada algoritmo avaliado no conjunto de validação é representado pelo regressor que obteve o melhor desempenho. Os modelos de regressão selecionados

a partir de cada algoritmo são comparados, e os melhores usados para construir o CB. Nesse sentido, é importante destacar que os modelos escolhidos são oriundos de algoritmos distintos, e que esta estratégia tem a intenção de garantir a diversidade do *ensemble*. Desta forma, terão sido selecionados os melhores modelos baseando-se no desempenho e na diversidade. O desempenho é medido através da média do erro absoluto de cada modelo no conjunto de validação, enquanto a diversidade é obtida pela heterogeneidade dos algoritmos, visto que cada regressor é oriundo de algoritmos distintos. Se os três melhores modelos de regressão - entre todos r regressores criados - forem oriundos do mesmo algoritmo, apenas o melhor deles deve ser escolhido, enquanto os outros modelos do conjunto básico deverão vir de algoritmos distintos, sendo escolhido aqueles que tiverem obtido o melhor desempenho.

A avaliação de desempenho dos modelos durante a segunda etapa é alcançada através dos seguintes métodos de validação: (i) reamostragem *hold-out* (mínimo de 30 iterações realizadas), sendo 2/3 dos dados de treinamento usados para criar os modelos (τ_r) e 1/3 dos dados de treinamento para validar (τ_v); (ii) validação cruzada *leave-one-out* ($k-fold = n$) para bases de dados pequenas ($n < 350$). O conjunto de validação apresentado na Figura 12 é extraído do conjunto de dados de treinamento inicial. Todos os modelos que forem significativamente inferiores ao melhor modelo de regressão avaliado na validação são eliminados, sendo ao menos três modelos escolhidos (apenas aqueles com os melhores desempenhos). A quantidade de regressores escolhidos não deve ser menor do que três, visto que um *ensemble* dinâmico com dois regressores resultará em um modelo muito próximo da seleção dinâmica simples. Ainda, se for necessário, ou seja, se o número de modelos estatisticamente equivalentes ao melhor modelo for maior que 3, é sugerida uma avaliação com todas as possíveis combinações três a três entre os regressores significativamente iguais ao melhor modelo, para que assim seja definido o CB mais adequado. Os três melhores modelos distintos é definido como o padrão do *framework* para compor o CB. A Figura 14 ilustra o processo de criação do CB.

Na terceira etapa, é realizada a integração dos modelos selecionados no CB, cuja fusão é dada pela média ponderada das saídas emitidas pelos regressores selecionados dinamicamente. Esses regressores são selecionados por classificadores treinados no Base de Treinamento dos Classificadores (τ_c), que é semelhante ao τ_r, que foi utilizado para gerar os regressores, todavia, apresentando a diferença entre si na variável dependente. Importante ainda lembrar que a τ_c tem os mesmos atributos de entrada da τ_r, assim como os seus respectivos valores. No entanto, na τ_c, o rótulo (variável dependente) indica o melhor regressor do CB avaliado para a respectiva instância de treinamento. O regressor do CB que obteve o menor erro na instância avaliada é rotulado nesse novo campo para a respectiva instância. Essa avaliação é realizada apenas na base de treinamento dos regressores, e não envolve os dados separados para validação, visto que esses serão usados posteriormente para validar os classificadores no processo de seleção dinâmica. A Figura

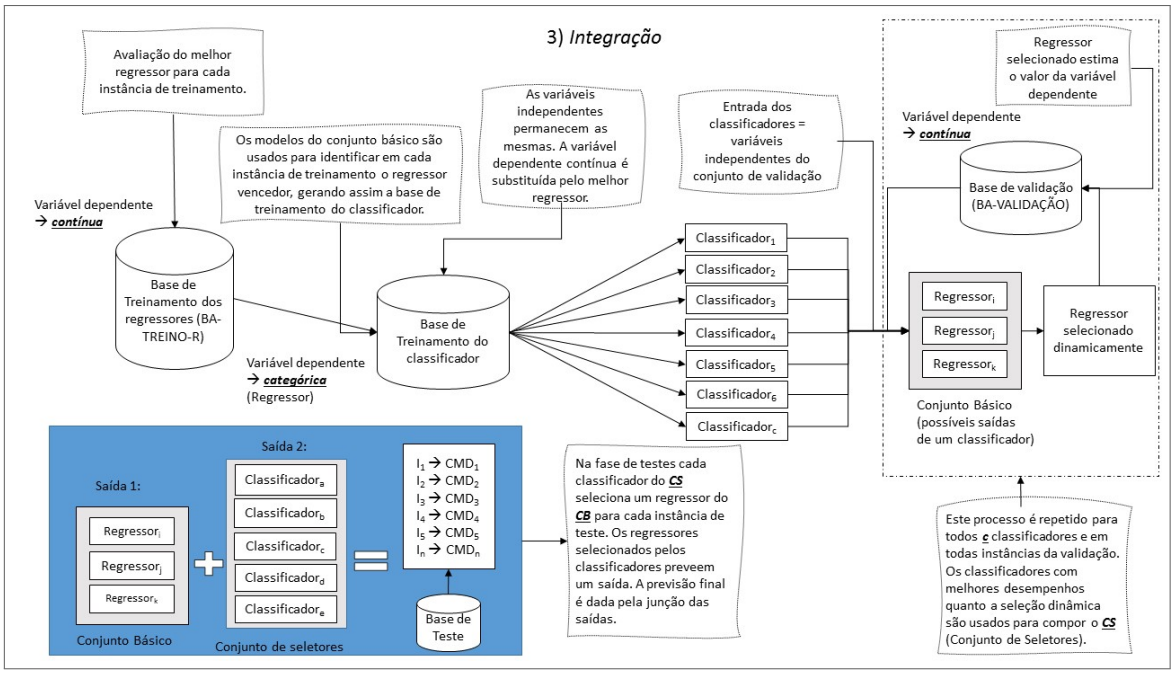
Figura 14 – Etapa de poda dos modelos de regressão do *Framework* proposto



Fonte: O autor (2022)

15 demonstra em detalhes esse procedimento.

Figura 15 – Etapa de integração dos modelos de regressão do *Framework* proposto



Fonte: O autor (2022)

O processo para identificar o melhor regressor para cada instância da τ_r pode ser feito inteiramente com a própria τ_r ou por validação cruzada. Utilizar a própria base para validar é mais rápido e experimentos realizados durante o estudo mostraram que essa

abordagem pode funcionar bem em alguns conjuntos de dados, mas esse tipo de estratégia não seria indicado, caso algum regressor do CB tenha a característica de decorar os dados de treinamento, por exemplo, um algoritmo *K-Nearest Neighbor* (KNN) com $k = 1$. Treinar os regressores do CB com a τ_r e comparar o desempenho deles com a própria τ_r poderá levar à conclusão de que o desempenho dos regressores que decoraram menos dados é pior e, de fato, não necessariamente corresponderá aos regressores mais fracos, tendo como consequência a possibilidade de diminuição do desempenho do método gerado a partir do *framework* proposto. No entanto, para bases de dados relativamente grandes, essa abordagem poderá alcançar bons resultados, conforme foi percebido em alguns testes prévios. Estudos posteriores poderão ser realizados, a fim de investigar esta hipótese.

Outro ponto a ser destacado é que, a fim de evitar um possível *overfitting*, deve-se utilizar validação cruzada na identificação dos melhores regressores para cada instância da τ_c, posto que essa opção é mais robusta e menos eficiente do que a anterior. No entanto, apesar do uso dela eliminar a possibilidade de *overfitting* no método, a depender da quantidade de partições (*k-fold*) e do tamanho da base de dados, o método gerado poderá se tornar custoso. Diante deste impasse, uma sugestão razoável é utilizar $k - fold = 2$ ou $k - fold = 5$. Esta é uma definição subjetiva que deve variar de acordo com a quantidade de dados e a experiência do especialista. Além do valor do *k-fold* usado na construção da τ_c, o *framework* também permite configurar o uso das probabilidades indicadas por cada classificador. Nesse sentido, essa variável deve ser investigada e definida na fase de validação. O uso desta opção permite gerar pesos dinâmicos para as saídas dos regressores. Quando esta opção é desabilitada, um regressor recebe peso 1 (o selecionado) e os demais peso 0 para cada instância avaliada por cada classificador.

Para exemplificar o que foi descrito, suponhamos que a base de treinamento inicial contenha 750 instâncias, as quais 500 (2/3) são usadas para gerar os modelos de regressão τ_r e 250 (1/3) para validação. Estas serão as instâncias que avaliarão os modelos de regressão criados na τ_r. A τ_c terá exatamente as mesmas 500 instâncias utilizadas para treinar os regressores na τ_r, no entanto, com a variável dependente modificada, agora, categorizada e identificada com um regressor. Dessa forma, os classificadores são criados a partir dos dados da τ_c e, em seguida, cada instância do conjunto de validação é dada como entrada para o classificador gerado, o qual dá como saída um modelo de regressão pertencente ao CB ou um conjunto de probabilidades para cada modelo. Por sua vez, o modelo de regressão selecionado ou as probabilidades de cada modelo serão usadas para estimar o valor da variável dependente da instância de validação corrente. Ao fim do processo de seleção e predição, os classificadores são avaliados de acordo com o desempenho dos regressores que cada um selecionou dinamicamente no conjunto de validação. Logo, o Conjunto de Seletores (CS) é definido pelo conjunto dos melhores classificadores (seletores) avaliados. A métrica usada para definir o CS pode variar e, portanto, sugere-se utilizar a média dos erros absolutos para dados com distribuição normal, e a mediana ou o *ranking*

para os dados que não assumirem uma distribuição normal.

Ao término da fase de validação, é escolhido um conjunto de classificadores CS para cada base de dados. A avaliação dos classificadores é individual e baseada no desempenho dos regressores selecionados por cada classificador na base de validação. A performance dos classificadores pode ser avaliada pela média ou mediana dos erros absolutos obtidos a partir das estimativas dos regressores selecionados dinamicamente na validação, sabendo-se que o número de regressores e de classificadores devem ser avaliado na validação. Em resumo, cinco hiper parâmetros do *framework* poderão ser configuráveis durante a fase de validação: (i) quantidade de regressores; (ii) a quantidade de classificadores; (iii) o valor do *k-fold* na construção da τ_c; (iv) a métrica de avaliação dos classificadores quanto aos regressores selecionados dinamicamente; e (v) o uso das probabilidades emitidas por cada classificador.

Na fase de testes, após a definição do CB (segunda etapa) e dos CS (terceira etapa), o processo anterior é repetido. Para cada instância de teste um classificador do CS seleciona um regressor do CB e, consequentemente, as estimativas de cada regressor são integradas e a saída é dada por um processo de fusão, em que as previsões de cada regressor selecionado dinamicamente são unidas através de uma regra de combinação (média, mediana). Por exemplo, suponhamos que cinco classificadores façam parte do CS e que três regressores estejam contidos no CB. Isso implica que a saída final se dará a partir das saídas dadas pelos regressores selecionados por intermédio dos cinco classificadores. Considerando que cada regressor pode ser selecionado mais de uma vez, ou até mesmo não ser selecionado, a função de saída utilizada para realizar a estimativa final será a média ponderada dinamicamente, visto que os pesos dados para cada regressor são calculados de acordo com a quantidade de vezes que ele é selecionado. Se a variável de uso das probabilidades estiver definida como verdadeira, o peso de cada regressor será definido a partir da média das probabilidades de cada regressor na saída de cada classificador. Perceba que o conjunto de classificadores usados para realizar a seleção dinâmica (CS) poderá variar de acordo com a métrica usada para realizar as avaliações na validação. Essas avaliações dar-se quanto ao desempenho dos regressores selecionados dinamicamente por eles. A Figura 15 apresenta o processo de integração dos modelos e destaca, com um fundo azul, a fase de teste.

Neste sentido, o *framework* proposto pode ser visto por duas abordagens distintas. A primeira aplica o conceito de *Dynamic Selection* (DS) simples, a qual está associada a seleção de um único modelo e, neste caso, apenas o melhor classificador é selecionado para compor o CS e prever o regressor ideal para cada instância de teste. Consequentemente, apenas um modelo de regressão será selecionado dinamicamente. Quando a variável do uso de probabilidades é verdadeira, mesmo com um classificador no CS, é gerado uma *Dynamic Ensemble Selection* (DES), visto que as probabilidades de saída do classificador são usadas como pesos para as saídas de cada regressor, Nesta opção já se aplica a segunda abordagem citada no início do parágrafo. A segunda abordagem também é construída se

mais de um classificador fizer parte do CS e, neste caso, com ou sem o uso das probabilidades dos classificadores. Tendo o CS mais de um classificador, mais de um regressor será selecionado.

Desta forma, no *framework* proposto, a agregação das saídas só será possível se ao menos dois classificadores forem escolhidos ou se um classificador for usado com as probabilidades ativas, caso contrário, a saída final será uma seleção dinâmica simples. Os experimentos realizados neste trabalho consideraram o uso de três regressores e cinco classificadores para todas as bases de dados. Quanto às demais configurações, foram verificados na validação os valores de $k-fold = 1$ e $k-fold = 2$, foi usada a média dos erros absolutos como métrica de desempenho padrão, e finalmente, foi desativada a variável de probabilidades.

O *framework* proposto difere das técnicas de seleção dinâmica do estado da arte não apenas porque usam múltiplos modelos de regressão heterogêneos, mas também porque a regra de seleção dos regressores é aprendida por diversos classificadores distintos. Nesse sentido, diferentes e rigorosos critérios são usados para selecionar regressores dinamicamente. Ainda, é importante mencionar que existem relatos na literatura que sugerem que a performance de diferentes modelos dependem consideravelmente das características dos conjuntos de dados (MINKU; YAO, 2013). Essa é uma afirmação que poderia ser extraída do conceito *No Free Lunch*, o qual foi mencionado no início desse trabalho. Nesse sentido, considerando que diferentes contextos de dados podem ser melhor ajustados para um ou outro algoritmo, foi definido através das bases de validação um conjunto específico de regressores e classificadores para cada base de dados investigada. Em resumo, um *ensemble* estático de regressores é ponderado dinamicamente por um *ensemble* estático de classificadores, de maneira que, modelos de regressão são dinamicamente selecionados e ponderados para prever o valor da variável dependente de uma base de dados.

4.3 IMPLEMENTAÇÃO

Nesta Seção será apresentado um passo a passo para exemplificar a utilização do *framework* proposto e as representações algorítmicas da implementação realizada.

4.3.1 Passo a passo

Os passos realizados para desenvolver e avaliar o *framework* proposto são detalhados abaixo, conforme segue:

1. Dividir a base de dados em treino, teste e validação.

   O método de validação usado (*hold-out*, *cross-validation* ou *leave-one-out*) foi definido de acordo com o tamanho do conjunto de dados. Para cada iteração, conjuntos de treino e teste foram extraídos da base de dados, sendo a validação parte do

conjunto de treinamento. Quando o *hold-out* foi usado ($n > 350$, n = tamanho da base), os dados são separados aleatoriamente para cada iteração (mínimo 30), com 75% do conjunto destinado para o treinamento e 25% para os testes, sendo 1/3 dos dados de treinamento utilizados para validação. Se a opção for pelas técnicas *cross-validation* ou *leave-one-out*, de maneira semelhante, uma partição é separada para os testes e outra para validação, sendo os demais dados são destinados ao treino. Esse procedimento se repete para cada iteração, de maneira que, ao final, todos os elementos tenham sido usados em alguma iteração para as fases de treino, teste e validação, mas em iterações distintas, conforme explicado na Figura 20.

2. Construir r modelos de regressão (rm_1, rm_2, ..., rm_r) a partir de n_a algoritmos.

   O conjunto de treinamento é usado para gerar os r modelos de regressão, enquanto o conjunto de validação para avaliá-los antes da fase de teste. Para cada iteração, é atribuído um erro absoluto médio para cada rm_e avaliado na validação. Os r modelos avaliados são comparados de acordo com a *Mean Absolute Error* (MAE), que define o desempenho de cada modelo rm. Dessa forma, na saída deste passo, cada modelo de regressão contém um conjunto de i médias de erros absolutos, sendo i a quantidade de iterações no experimento.

3. Selecionar rm diversos baseando-se no desempenho.

   De acordo com os resultados prévios da fase de validação, os melhores modelos de regressão são selecionados. Esses modelos são escolhidos a partir do desempenho de cada um, de modo que o melhor modelo de cada algoritmo é escolhido. Desta forma, é alcançada a diversidade de modelos, a qual pode ser encontrada nos *ensembles* heterogêneos.

4. Avaliar os rm usando testes estatísticos.

   Os resultados alcançados por cada modelo escolhido no passo anterior foram comparados através de testes estatísticos. Nesse sentido, inicialmente, cada modelo é avaliado na base de validação. Assim, para cada iteração teremos uma MAE para cada modelo, e as i iterações resultarão em i MAE para cada modelo. A partir desses erros, um teste estatístico adequado, Análise de Variância (ANOVA) (distribuição normal) ou o teste de *Friedman* (não normal), deve ser usado a fim de identificar os melhores modelos de regressão. São selecionados os modelos que foram significativamente semelhantes ao melhor modelo avaliado ou, por definição padrão do *framework*, simplesmente os três melhores modelos. Se a primeira opção for escolhida e a quantidade de modelos semelhantes ao vencedor for maior ou igual a três, deve ser realizada uma segunda avaliação com todas as possíveis combinações dos modelos que obtiveram resultados semelhantes ao modelo vencedor. Por padrão, são escolhidos os três melhores modelos, ficando a critério do pesquisador realizar testes

extras na validação. Consequentemente, após a definição dos melhores modelos na fase de validação, é construído o CB.

5. Definir o melhor rm do CB para cada instância da τ_r ($\tau_{r_{\vec{1}}}$, $\tau_{r_{\vec{2}}}$, $\tau_{r_{\vec{3}}}$, $\tau_{r\vec{4}}$, $\tau_{r_{\vec{5}}}$ ... $\tau_{r_{\vec{i}}}$) e adicionar um novo campo para identificá-lo.

   Neste passo, um conjunto de dados para problemas de classificação chamado τ_c, que é construído a partir da τ_r, é criado. As instâncias da τ_r são avaliadas por validação cruzada ou com a própria base de treinamento. Conforme foi visto, as implicações de cada forma de avaliação foram discutidas na seção anterior. Um novo campo é adicionado para identificar o melhor regressor entre os modelos individuais que pertencem ao CB e, nesse sentido, um rótulo identificando o melhor regressor é o atributo dependente da τ_c, sempre lembrando que os atributos independentes são os mesmos da τ_r. Perceba que esse processo ocorre em cada iteração do experimento, de maneira que, se tivermos i iterações, teremos i τ_c criadas, uma para cada iteração do experimento. A Tabela 2 ilustra o que foi descrito.

6. Treinar c classificadores usando as τ_c.

   Para cada iteração, uma τ_c é usada para treinar classificadores aptos a realizar a seleção dinâmica de regressores. Um conjunto de c classificadores são induzidos a partir τ_c, os quais associarão o conjunto de atributos de entrada com um modelo de regressão pertencente ao CB na validação, e posteriormente, nos testes.

7. Validar os c classificadores.

   Uma vez treinado vários classificadores, é iniciado o procedimento que irá definir modelos diversos baseado no desempenho. Os classificadores selecionados neste passo devem ser oriundos de algoritmos distintos. A seleção é baseada nos erros de estimativas dos regressores selecionados pelos classificadores para cada instância na validação. Dessa forma, para cada instância de validação, um classificador sugere um modelo de regressão do CB para estimar o valor da variável dependente dessa instância, processo que é repetido para cada iteração. Ao final, cada classificador receberá um conjunto de MAE obtido a partir das estimativas dos regressores que ele selecionou para cada instância da base de validação.

8. Selecionar os classificadores do cs.

   Os melhores classificadores são escolhidos, baseado no desempenho alcançado pelos modelos de regressão selecionados dinamicamente por eles. A comparação dos resultados pode ser realizada a partir da média, da mediana ou do ranking. Se a distribuição dos erros tender a uma normalidade, devem ser escolhidos os classificadores que alcançarem o melhor desempenho na seleção quanto à média dos erros, caso contrário, é indicado que seja usado a mediana ou o *ranking* para definir os classificadores do CS. Cada conjunto de dados foi associado a um cs, e os classificadores

escolhidos neste passo foram usados para selecionar regressores dinamicamente na base de teste.

9. Usar o CS para selecionar os regressores do CB para cada instância de teste.

   Neste passo, inicia-se o processo de teste. Os classificadores escolhidos no passo anterior são usados para selecionar dinamicamente regressores do CB. Os modelos de classificação recebem um conjunto de atributos independentes de cada instância de teste, e dão como saída um regressor do CB que, ao ser selecionado estima o valor da variável dependente da instância de teste corrente. Desta maneira, é possível que os regressores selecionados dinamicamente se repitam para classificadores diferentes. Consequentemente, pesos dinâmicos são gerados para cada regressor, e o tamanho do CS definirá a quantidade de regressores selecionados. No entanto, é possível ativar as probabilidades de saída dos classificadores, para que assim os pesos da saída final de cada regressor sejam proporcionais a essas probabilidades. Com as probabilidades ativas, o uso de apenas um classificador poderá selecionar mais de um regressor através de pesos dinâmicos que são definidos pelas probabilidades de cada regressor. Podemos perceber que, na fase de testes, temos um conjunto de modelos de regressão (CB) e de classificação (CS) que foram definidos na validação.

10. Agregar as saídas dos regressores do CB selecionados pelos classificadores do CS.

    A estimativa final é dada pela média ponderada das saídas dos modelos de regressão selecionados pelos classificadores do CS. Quando é usado apenas o classificador mais bem avaliado na validação e as probabilidades estão desativadas, é realizada seleção dinâmica simples, que chamaremos de *Process of Evaluating Estimates Through Algorithm Combinations - Dynamic Selection* (PEETACO-DS); entretanto, quando são usados vários classificadores, $n_c > 1$, ou as probabilidades de saída estão ativadas, são selecionados n_c modelos de regressão com pesos dinâmicos para realizar a estimativa final de cada instância de teste. A esse tipo de método proposto nomearemos *Process of Evaluating Estimates Through Algorithm Combinations - Dynamic Ensemble Selection* (PEETACO-DES).

De acordo com as fases de construção de um SMM, conforme apresentado no Capítulo 2, percebe-se que a fase de geração de modelos é coberta do passo 1 ao passo 3. Em seguida, temos a fase de poda que é iniciada e completada no passo 4 e, finalmente, a fase de integração, incluindo os testes, que vai do passo 5 ao 10.

Ao fim de todos os passos, foram avaliados os resultados dos métodos investigados nas bases de dados de testes. Nesse sentido, foram utilizados os resultados alcançados por cada método, desde os individuais até as diferentes combinações entre eles, incluindo os métodos de seleção dinâmica criados a partir do *framework* proposto neste trabalho. Por

fim, foi lançado mão também de testes estatísticos adequados para cada experimento, os quais foram mencionados no Item 4.4.4.

Tabela 2 – Exemplo ilustrativo das bases de treinamento dos regressores e dos classificadores

Ilustração da base de treinamento dos regressores						
Instância	Atr. 1	Atr. 2	Atr. 3	Atr. 4	Atr. 5	Saída Real
1	Sim	10	1	A	Sim	100
2	Sim	20	5	B	Sim	250
3	Não	30	3	B	Não	230
4	Não	40	3	C	Sim	320
5	Não	50	3	A	Não	250
Estimativas e erros dos modelos						
Instância	Saída A	Saída B	Saída C	Resíduo A	Resíduo B	Resíduo C
1	110	111	130	10	11	30
2	235	257	260	15	7	10
3	200	233	255	30	3	22
4	315	350	323	5	30	3
5	230	255	260	20	5	10
Ilustração da base de treinamento dos classificadores						
Instância	Atr. 1	Atr. 2	Atr. 3	Atr. 4	Atr. 5	Modelo
1	Sim	10	1	A	Sim	Modelo A
2	Sim	20	5	B	Sim	Modelo B
3	Não	30	3	B	Não	Modelo B
4	Não	40	3	C	Sim	Modelo C
5	Não	50	3	A	Não	Modelo B

Fonte: O autor (2022)

4.3.2 Representação algorítmica

Nesta seção, apresentaremos três algoritmos que descrevem o *framework* proposto. O algoritmo 2 apresenta de forma estruturada o procedimento para obter os regressores que compõe o CB, enquanto o algoritmo 3 apresenta a estrutura para obter o conjunto de classificadores que serão usados na seleção dinâmica. Finalmente, o algoritmo 4 ilustra o processo de testes.

Algoritmo 2: PEETACO - Conjunto de regressores (CB)

Entrada: τ_r, τ_v;

1: r = *tamanho do conjunto de regressores individuais*;
2: *Treine usando os dados de* τ_r *e induza r regressores,* $R = \{R_1, R_2...R_r\}$;
3: $m = 1$; $j = 1$; $i = 1$;
4: **para** cada iteração i **faça**
5: **para** cada modelo de regressão R_m até R_r **faça**
6: **para** cada instância de validação τ_{v_j} **faça**
7: $\hat{R_{m_j}}$ = predição do modelo R_m em j;
8: $e_j = |\mathrm{y}_j - \hat{\mathrm{R}_{\mathrm{mj}}}|$;
9: $Errors_{R_m} = Errors_{R_m} + e_j$;
10: $j = j + 1$;
11: **fim do para**
12: $mae = mean(Errors_{R_m})$
13: $listErrors_{R_{m_i}} = add(mae)$
14: $m = m + 1$;
15: $j = 1$;
16: **fim do para**
17: $listErrors_{R_m} = add(listErrors_{R_{m_i}})$
18: $i = i + 1$
19: **fim do para**
20: *// Avaliação de diversidade dos r modelos individuais.*
21: $diversesRegressors$ = TheBestByAlgorithm ($listErrors_{R_{1_1}}$,..., $Errors_{R_{r_i}}$)
22: *// Avaliação estatística do desempenho dos r regressores diversos.*
23: CB = TheThreeBest($StatisticalTest$, $diversesRegressors$)
24: *// regressores que formam o* $CB = \{R_{m_1}, R_{m_2}, R_{m_3}\}$
25: **retorne** CB

Algoritmo 3: PEETACO - Conjunto de classificadores (CS)

Entrada: CB, τ_r, τ_v;

1: $it = 1$;
2: *// criação da base de treinamento dos classificadores*
3: **para** cada instância de treino τ_{it} **faça**
4: $RM_{chosen} = MinimumErrorEvaluated\ (\tau_{it}, CB)$;
5: $RM_{chosen} = rm_{m_1}$ ou rm_{m_2} ... rm_{m_3}
6: $nit = replaceTarget(\tau_{it}, RM_{chosen})$; *// nova instância*
7: $\tau_c = \text{add}(nit)$; *// adiciona instância na base de classificação*
8: $it = it + 1$;
9: **fim do para**
10: n_c = *quantidade de classificadores*;
11: *Utilize os dados de* τ_c *para induzir* n_c *modelos de classificação*
12: $C = \{c_1, c_2 ... c_{n_c}\}$;
13: $m = 1$; $j = 1$; $i = 1$;
14: **para** cada iteração i **faça**
15: **para** cada modelo de classificação C_m **faça**
16: **para** cada instância de validação VA_j **faça**
17: $R_s = \text{DynamicSelectionRegressor}(\tau_{v_j}, C_m, CB)$;
18: $\hat{C_m}$ = predição do modelo R_s em j;
19: $error_j = |\text{y}_j - \hat{\text{C}_{\text{m}_\text{j}}}|$;
20: $Errors_{C_m} = Errors_{C_m} + e_j$;
21: $j = j + 1$;
22: **fim do para**
23: $mae = mean(Errors_{C_m})$
24: $listErrors_{C_{m_i}} = add(mae)$
25: $m = m + 1$;
26: $j = 1$;
27: **fim do para**
28: $listErrors_{C_m} = add(listErrors_{C_{m_i}})$ $i = i + 1$;
29: **fim do para**
30: *Avalie* n_c *modelos de classificação e escolha aqueles que obtiveram o melhor desempenho de acordo com* $listErrors_{C_m}$.
31: $diversesClassifiers = \text{TheBestByAlgorithm}\ (listErrors_{C_{1_1}}, ..., Errors_{C_{nc_i}})$
32: $CS = \{C_{m_1}, C_{m_2}, C_{m3}, C_{m4}, C_{m5}\}$
33: retorne CS

Algoritmo 4: PEETACO - Teste

Entrada: τ_t, CS, CB;

1: **para** cada instância de teste τ_{it} **faça**
2: *// considerando* n_c *classificadores*
3: **para** cada classificador de CS_{mc} até n_c **faça**
4: $R_s = \text{DynamicSelectionRegressor}(\tau_{it}, CS_{mc}, CB)$;
5: $\hat{En} = \hat{En} + \hat{R}_s$
6: **fim do para**
7: $\hat{PEETACO} = \hat{En}/n_c$
8: **fim do para**

4.4 AVALIAÇÃO

Nesta seção, serão apresentados os métodos usados para fins de comparação e avaliação do *framework* proposto e, para isso, foram usados algoritmos comumente abordados em problemas de regressão. Além desses, métodos de seleção de *ensembles* estáticos e dinâmicos também foram incluídos, a fim de fortalecer a apresentação dos resultados.

4.4.1 Algoritmos de regressão e classificação

Os testes realizados abordaram 14 algoritmos de regressão e 19 de classificação, tendo sido cada um deles individualmente avaliado na validação. Para cada conjunto de dados, foi formado um CB e um CS. A escolha e a quantidade de algoritmos foram definidas de acordo com a disponibilidade da biblioteca *Weka 3.6.10* e dos resultados apresentados nos estudos (WEN et al., 2012; IDRI; HOSNI; ABRAN, 2016b; IDRI; HOSNI; ABRAN, 2016a; ROMERO; VENTURA, 2010a; ROMERO; VENTURA, 2013).

A fim de diversificar os SMM gerados, foram selecionados, desde algoritmos simples, como o ZeroR, que simplesmente prever o valor da nova instância a partir da média da variável dependente nas instâncias de treinamento, até as redes neurais *Multi Layer Perceptron* (MLP) (BRAGA; CARVALHO; LUDERMIR, 2007). Quanto aos algoritmos de classificação, métodos baseados em árvores, regras e distâncias foram os mais comuns, além desses, a regressão logística e o *Support Vector Machine* também fizeram parte dos algoritmos investigados.

A Tabela 3 apresenta a lista de algoritmos de regressão e classificação avaliados neste trabalho. É apresentado o nome de cada algoritmo, em seguida, entre parênteses, as iniciais usadas para referenciá-lo. Outrossim, é informado o grupo que o algoritmo pertence, o tipo de problema que ele foi usado e, finalmente, a referência. Alguns algoritmos têm como referência a própria página de documentação da biblioteca *Weka*, uma vez que eles não são literalmente baseados em um estudo específico da literatura.

A documentação completa da biblioteca pode ser encontrada em (WAIKATO, 2022h). Os valores dos hiper parâmetros de cada algoritmo foram definidos de acordo com as variáveis disponíveis na biblioteca (*Weka 3.6.10*). Cada algoritmo foi representado por no mínimo uma e no máximo nove configurações diferentes. No Apêndice B, é apresentado o código em linguagem Java referente à geração dos modelos citados na Tabela 3. No código, é possível perceber que cada configuração foi identificada pelas iniciais do algoritmo e por um número. Consequentemente, essas configurações possuem uma única hiper parametrização do algoritmo, sendo identificadas de maneira única por todo o trabalho. Os hiper parâmetros não modificados permaneceram com os valores padrões da biblioteca.

Tabela 3 – Lista de algoritmos de regressão e classificação usados no trabalho

Algoritmo (Iniciais) - Grupo (Tipo) - Referência
Adaboost (AD) - *Ensembles* (Class.) - (FREUND; SCHAPIRE, 1996)
Additive Regression (AR) - *Ensembles* (Reg.) - (FRIEDMAN, 2002)
Bagging (BA) - *Ensembles* (Reg./Class.) - (BREIMAN, 1996)
Best First Tree (BFT) - Árvores (Class.) - (SHI, 2007)
Conjunctive Rules (CR) - Regras (Reg.) - (WAIKATO, 2022a)
Decision Table (DT) - Regras (Reg.) - (KOHAVI, 1995)
Gaussian Process (GP) - Funções (Reg.) - *FilterType* e *Kernel* - (MACKAY, 1998)
K-Nearest Network (KNN) - Distância (Reg./Class.) - (AHA; KIBLER; ALBERT, 1991)
J48 (J48) - Árvores (Class.) - (QUINLAN, 1993)
KStar (KS) - Distância (Class.) - (CLEARY; TRIGG, 1995)
Least Median Squared (LMS) - Funcão (Reg.) - (ROUSSEEUW; LEROY, 1987)
Linear Regression (LR) – Funções (Reg.) - (WAIKATO, 2022b)
Logistic Model Trees (LMT) - Árvores (Class.) - (LANDWEHR; HALL; FRANK, 2005)
Locally Weight Least (LWL) - Distância (Reg./Class.) - (ATKESON; MOORE; SCHAAL, 1997)
Logistic Regression (LoR) - Funções (Class.) - (WAIKATO, 2022c)
M5 Base (M5P) - Árvores (Reg.) - (QUINLAN, 1992)
M5 Rules (M5R) - Regras (Reg.) - (WANG; WITTEN, 1996)
Multilayer Perceptron (MLP) - Funções (Reg./Class.) - (WAIKATO, 2022d)
Naive Bayes (NB) - Probabilísticos (Class.) - (JOHN; LANGLEY, 2013)
Oner (OR) - Regras (Class.) - (HOLTE, 1993)
Random Forest (RF) - Árvores (Class.) - (BREIMAN, 2001)
RBF Network (RBF) - Funções (Reg.) - (WAIKATO, 2022e)
Rep Tree (RT) - Árvores (Reg./Class.) - (WAIKATO, 2022f)
Support Vector Machine (SVM) - Funções (Class.) - *KernelType* -(PLATT, 1998)
Support Vector Regression (SVR) - Funções (Reg.) - (SHEVADE et al., 2000)
Zero R (ZEROR) - Regras (Reg.) - (WAIKATO, 2022g)

Fonte: O autor (2022)

4.4.2 Métodos de avaliação

De maneira geral, a comparação básica realizada para um SMM analisa a melhoria alcançada em relação aos modelos bases (homogêneos), ou individuais (heterogêneos). Ou seja, dentro do contexto deste trabalho, os métodos de avaliação serão os *baselines*, que irão verificar a eficácia da proposta em relação aos modelos individuais que compõe o CB. Ainda, é comum que essa avaliação seja realizada apenas com o melhor modelo individual na fase de validação, muito embora os resultados apresentados neste trabalho tenham abordado todos os modelos individuais pertencentes ao CB.

Além de comparar com os modelos individuais, sistemas de múltiplos modelos também foram adicionados na lista de métodos de avaliação. Uma estratégia simples para construir um SMM é unir as saídas dos modelos individuais (*ensembles* heterogêneos) ou dos modelos bases (*ensembles* homogêneos) e estimar o resultado a partir de funções de agregação. Neste sentido, as saídas individuais do CB também foram agregadas através de funções lineares (média, mediana, média entre extremos, mínimo e máximo). Além desses métodos de seleção estática, o *Stakcing* também foi adicionado ao grupo de *ensembles* heterogêneos, enquanto os métodos *Bagging* e *Boosting* foram adicionados ao grupo de *ensembles* homogêneos. Cada regressor pertencente ao CB foi usado como modelo base do *Bagging* e do *Boosting*. Vale lembrar, em contra partida, que todas estas estratégias de combinação não são dinâmicas, e podem nos levar a falhas na combinação dos modelos (KOCAGUNELI; MENZIES; KEUNG, 2012; KITTLER et al., 1998). Nesse sentido, a fim de comparar os resultados obtidos com os modelos gerados a partir do *framework* proposto, cinco métodos da literatura de seleção dinâmica foram empregados no estudo experimental deste trabalho. São eles: *Dynamic Classifier Selection By Local Accuracy* (DCS-LA), *Dynamic Classifier Selection By Local Accuracy Weighted* (DCS-LAW), *Author of the Adaptive Selection of Classifiers in Bug Predicting* (AASC-NUCCI) *K-Nearest Oracle Eliminate* (KNORA-E) e *K-Nearest Oracle Union* (KNORA-U). Estes métodos foram abordados nos Itens 2.2.4 e 2.2.5.

Como pode ser observado em (BRITTO; SABOURIN; OLIVEIRA, 2014), é impossível definir o melhor método de seleção dinâmica, uma vez que não existem evidências que uma técnica específica supere todas as outras em qualquer tarefa de aprendizagem, porém, o estudo mostra que os métodos DCS-LA e *K-Nearest Oracle* (KNORA) têm alcançado resultados similares em diferentes tipos de problemas e que eles são adotados em vários trabalhos da literatura. Além do DCS-LA e suas variações, também foi investigado outro método de seleção dinâmica simples proposto por (Di Nucci et al., 2017), o qual é capaz de selecionar dinamicamente o classificador que melhor prevê a possibilidade de falhas de uma Classe baseada em suas características. Este método foi adaptado para o contexto deste trabalho, uma vez que ele não foi usado para problemas de regressão.

Ainda em relação aos métodos de seleção dinâmica, é importante destacar determinadas particularidades existentes em alguns deles, como, por exemplo, no KNORA, uma

vez que a estratégia de seleção usada nesse método (BRITTO; SABOURIN; OLIVEIRA, 2014) é baseada em problemas de classificação. Neste sentido, a estratégia de seleção dinâmica aplicada ao KNORA precisou ser ajustada para prever uma variável contínua, visto que um conjunto de modelos selecionados para problemas de classificação não podem ser aplicados diretamente para problemas de regressão. Semelhante ao KNORA, o DCS-LA também foi projetado para problemas de classificação. De forma análoga, foi preciso realizar ajustes no processo de validação dos modelos individuais selecionados pelos métodos DCS-LA e DCS-LAW a fim de habilitá-los a tratar problemas de regressão.

A adaptação do DCS-LA para problemas de regressão foi baseada na seleção do regressor que obteve o menor erro médio entre os vizinhos mais próximos (k=*3* / k=*7*). Foram investigados dois valores de k para a região de competência, cujo tamanho pode afetar os resultados e é um problema a ser estudado (MOREIRA et al., 2009). Na abordagem original, o classificador é obtido baseado na melhor acurácia ou precisão da classe de saída. O Item 2.2.4 abordou esse método com mais detalhes. Para o KNORA, foi considerado que um modelo de regressão generaliza bem para um dos vizinhos mais próximo, quando o erro de estimativa na instância vizinha é menor do que 50% do valor real. Semelhante ao DCS-LA, dois valores de k foram avaliados dentro do contexto do KNORA, em que, baseado nessa estratégia, um modelo é adicionado (KNORA-U) ou eliminado (KNORA-E) do conjunto de modelos gerados dinamicamente. A seção 2.2.5 apresenta detalhes do KNORA.

Por fim, as avaliações do *framework* proposto foram realizadas através das comparações dos erros de estimativas obtidos pelos modelos individuais, além de sistemas de múltiplos modelos, que utilizaram seleção estática ou dinâmica. A seguir são listados os métodos concorrentes em mais detalhes.

1. Seleção estática de um modelo individual;

2. Seleção estática de múltiplos modelos heterogêneos. O CB foi usado e as saídas foram combinadas com métodos lineares, além do *Stacking* usando *Support Vector Regression* (SVR) como meta-regressor e os modelos do CB como regressores;

3. Seleção estática de modelos individuais aplicados em *ensembles* homogêneos, usando o *Bagging* e *Boosting*. Foram considerados os modelos individuais do CB como modelos bases dos *ensembles* homogêneos;

4. Métodos de seleção dinâmica simples (DCS-LA, DCS-LAW, e AASC-NUCCI), aplicados ao CB;

5. Métodos de seleção dinâmica de múltiplos modelos (KNORA-E e KNORA-U) aplicados ao CB.

As principais motivações para definir esses métodos como *baselines* foram: (i) a identificação de vários trabalhos da literatura que obtiveram bons resultados, (ii) o progresso que

os sistemas de múltiplos modelos têm alcançado, e (iii) as vantagens de utilizar métodos de seleção dinâmica. O resultado das comparações é apresentado no Capítulo 6.

4.4.3 Técnicas de validação

Um modelo é considerado bom se ele conseguir alcançar bom desempenho para novos dados de entrada do domínio do problema, ou seja, uma melhor capacidade de generalização (MITCHELL, 1997). Para verificar o quão bem um modelo aprende e generaliza, é preciso definir uma função Objetivo, logo, o aprendizado busca minimizar os erros da função objetivo. A minimização dos erros nos dados conhecidos (treinamento) e consequentemente nos dados desconhecidos (testes) é um desafio que requer o ajuste ideal do modelo. O ajuste excessivo do modelo aos dados pode gerar o *overfitting* (superajuste) do modelo, enquanto o uso de modelos muito simples pode gerar o *underfitting* (subajuste).

Um modelo é considerado super ajustado quando tende a memorizar ou capturar tendências e comportamentos dos dados. É importante que um modelo seja robusto quanto à presença de ruídos e a entradas irrelevantes pois, esses dados podem fazer com que os modelos sejam afetados. Existem diversas propostas para selecionar o modelo mais adequado para cada problema. Estratégias usadas para se evitar *overfitting* são: a re-amostragem dos dados e a validação cruzada, já que, nelas, os modelos aprendem sobre subamostras, a fim de estimar os erros inerentes ao processo de aprendizagem. Basicamente, essas técnicas buscam testar como o modelo se comporta diante de dados desconhecidos. Enquanto o *overfitting* indica que o modelo se ajustou muito aos dados de treino, a ocorrência de *underfitting* significa que o modelo não se adequou bem aos dados. Este tipo de problema normalmente ocorre quando a quantidade de dados é pequena para construir um modelo preciso ou quando modelos lineares são utilizados para dados não lineares. É comum aumentar a quantidade de dados e diminuir a dimensionalidade do problema, quando possível, para evitar modelos não generalizáveis.

A re-amostragem de dados através da bases de treino e testes geradas aleatoriamente e a validação cruzada são maneiras de aproximar o desempenho médio do modelo naquele problema. Essas técnicas de reamostragem não permitem que o modelo não sofra *overfitting*. Um bom desempenho na validação indica que o modelo tende a ser robusto em dados de testes. Um modelo ruim na validação e bom com dados de treino indica que ele possa estar super ajustado aos dados. A seguir são descritas as três técnicas de validação utilizadas neste trabalho. A eficácia de cada técnica está relacionada ao custo da implementação.

1. *Hold-out*:

   Esse método atribui aleatoriamente pontos de dados para dois grupos, treino e teste. Normalmente, o tamanho do grupo de treino é maior do que o de teste e as relações mais comuns são de 70% a 80% para treino e de 30% a 20% para testes, respec-

tivamente. O desempenho de cada modelo no grupo de validação vai indicar como ele deverá se comportar diante de dados novos. Dessa maneira o modelo tende a não ficar super ajustado aos dados de treinamento. Esta métrica recebe críticas pelo fato de que ela subestima a taxa de acerto, visto que apenas parte dos dados são testadas. Além disso, o método não permite avaliar quanto o desempenho de uma técnica varia quando diferentes combinações de objetos são apresentadas (FACELI et al., 2011). Nesse sentido, uma maneira de solucionar as críticas e melhorar o desempenho do modelo criado é repetir o processo de Treino-Teste n vezes e obter um média de desempenho; essa estratégia é por vezes conhecida como *random subsampling*. A cada iteração, o conjunto de dados de treinamento e de teste são construídos de maneira aleatória, o que leva a n bases de treinamento e teste distintas. A Figura 16 apresenta um modelo Treino-Teste (*Hold-out*) tradicional. Quando é necessário ajustar os hiper parâmetros do modelo ou comparar diferentes modelos, uma estratégia comum é separar parte dos dados de treino em um grupo de dados de validação, conforme a Figura 17.

2. *K-Fold Cross Validation*:

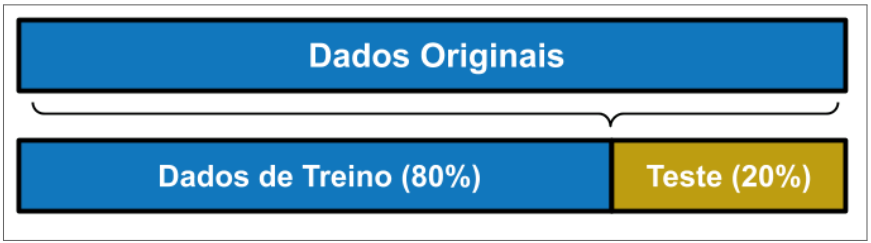
   Nessa reamostragem, a amostra original é dividida aleatoriamente em k subamostras de tamanhos iguais (FACELI et al., 2011). Dessas subamostras, uma é mantida como dados de teste para mensurar a qualidade do modelo gerado a partir das $k-1$ subamostras restantes, usadas como dados de treinamento, e o processo de validação cruzada é repetido k vezes, sendo cada uma das subamostras usada exatamente uma vez como dados de teste. A Figura 18 apresenta um esquema de validação cruzada com $k = 5$. A média dos erros de cada grupo pode ser usada para definir a performance do modelo. Semelhante ao caso anterior, quando é necessário ajustar os hiper parâmetros dos modelos ou fazer comparações, um esquema construído com um grupo de dados de validação para avaliar os modelos costuma ser usado, conforme a Figura 19.

3. *Leave-one-out*:

   Também conhecido como *jacknife*, o método computa n subconjuntos, sendo n o tamanho da amostra de dados. Um elemento de cada amostra é eliminado para fins de testes e, assim, cada amostra tem o tamanho de $n-1$ dados de treino e 1 dado para teste(FACELI et al., 2011). O processo é repetido até que todos os exemplos sejam usados como um dado do teste, fazendo com que todos os dados são usados para validar os modelos. O método é o mais robusto dos três apresentados, no entanto, requer mais recursos, por ser uma implementação mais cara e, por essa razão, pode ser visto como uma validação cruzada com $k = n$. Nesse sentido, a Figura 18 poderia ser usada para representar o *leave-one-out* em uma base de dados com 5 elementos, lembrando que uma base de dados com n elementos conteria n iterações. A Figura

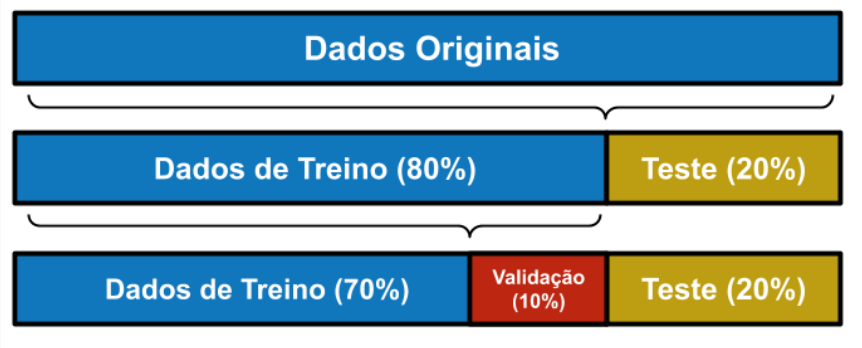
20 apresenta uma validação cruzada aninhada, em que um grupo de dados é usado para teste, enquanto os demais para, treino e validação. Porém, difere da técnica anterior porque a base de teste não é fixada no início do processo, de maneira que todos os dados, de alguma forma, são usados para testar os modelos.

Figura 16 – *Hold-out* tradicional



Fonte: (SCACCIA, 2020)

Figura 17 – *Hold-out* com validação



Fonte: (SCACCIA, 2020)

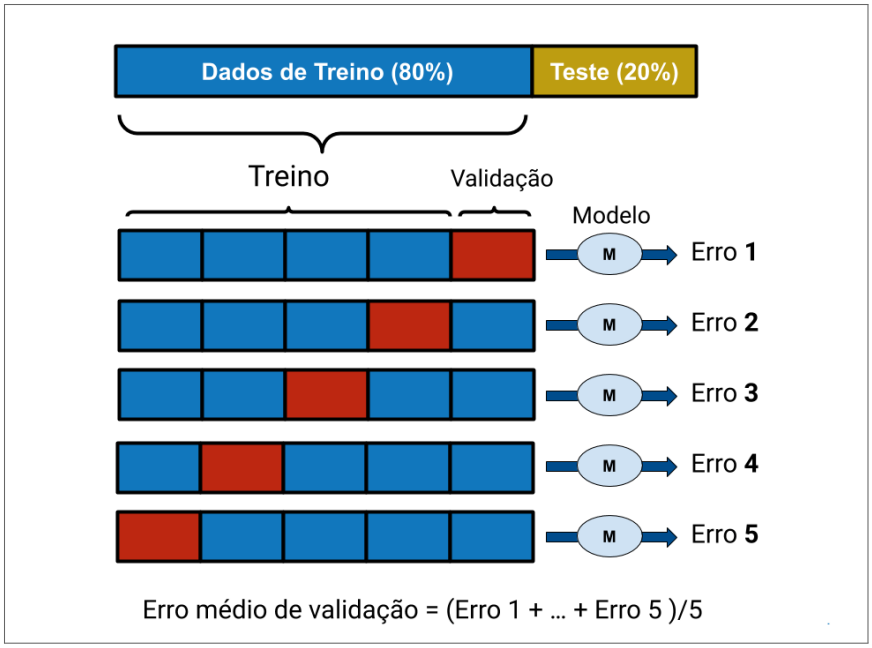
Quanto aos métodos de validação dos modelos usados neste estudo, as seguintes estratégias foram consideradas: (i) base de dados com mais de 350 exemplos foram validadas usando *Hold-Out* com reamostragem (30-50 iterações), enquanto *Leave-One-Out* foi a técnica escolhida para as bases de dados com menos de 350 exemplos. A técnica *K-Fold Cross Validation* foi usada para avaliar os modelos do CB na construção da τ_c.

De acordo com o que foi apresentado nessa seção, é perceptível que os modelos buscam minimizar uma determinada métrica de erro para um conjunto de dados avaliado. A seguir são apresentadas as métricas usadas em problemas de regressão.

4.4.4 Métricas e testes estatísticos

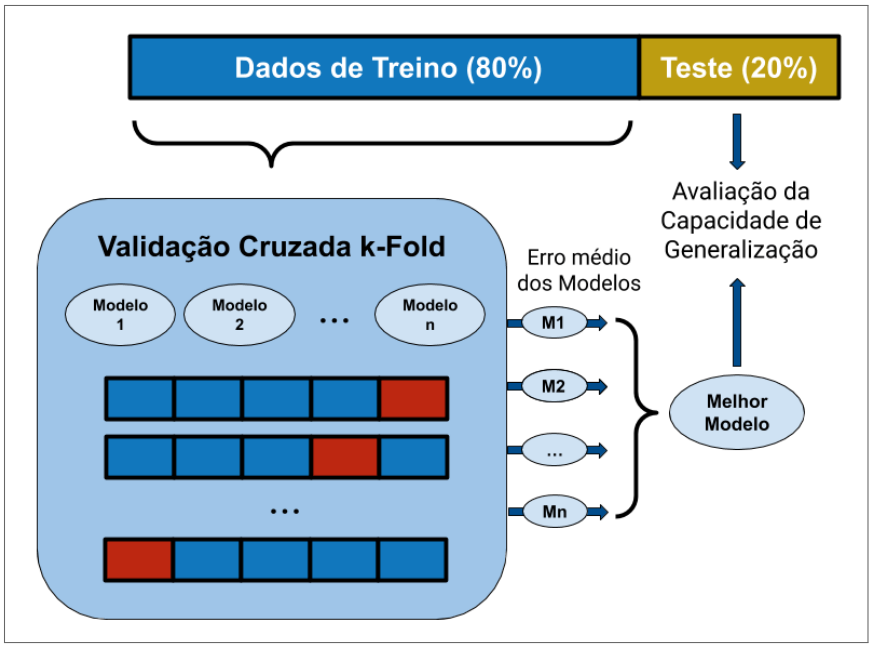
Os experimentos realizados permitiram selecionar, ao fim do processo de validação, três algoritmos de regressão e até cinco de classificação. Com o propósito de aumentar a acurácia dos modelos sugeridos, a seleção foi baseada na acurácia alcançada pelos métodos individuais pertencentes ao CB. Além desses, os melhores modelos de classificação (CS), de acordo com o processo de validação, foram selecionados para cada base de dados.

Figura 18 – Validação cruzada com $k = 5$



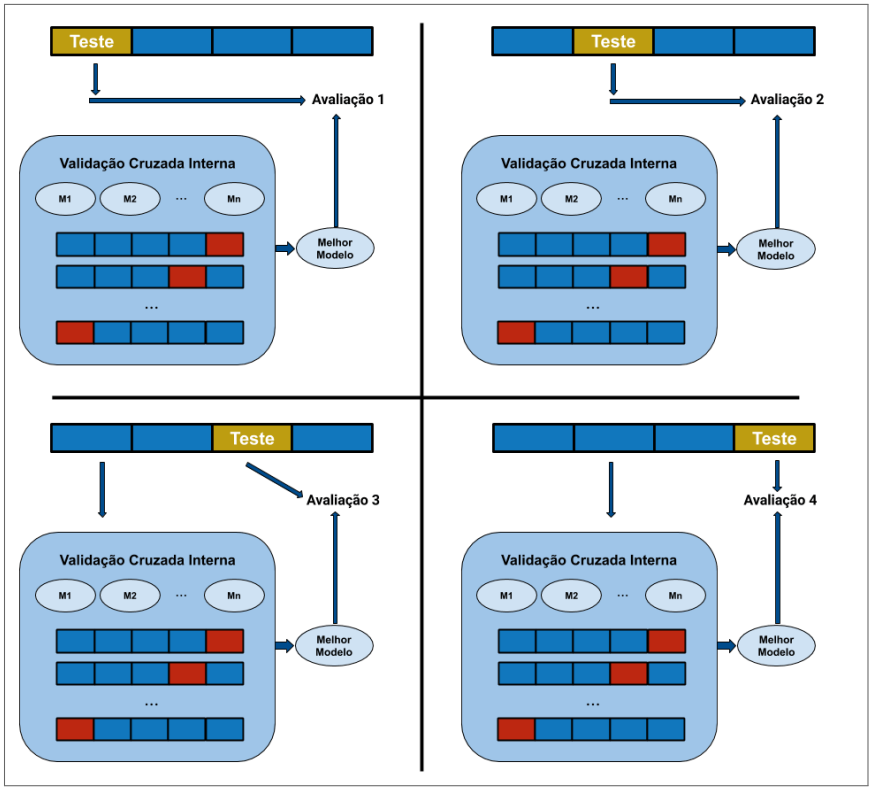
Fonte: (SCACCIA, 2020)

Figura 19 – Validação cruzada com comparação e seleção de modelos



Fonte: (SCACCIA, 2020)

Figura 20 – Validação cruzada aninhada com teste



Fonte: (SCACCIA, 2020)

Diante disso, é preciso definir as métricas para avaliação dos modelos, a fim de comparar os resultados. A seguir apresentamos as cinco métricas de avaliação utilizadas nos experimentos.

1. Média dos erros absolutos: é a métrica mais simples usada em problemas de regressão, do inglês MAE - *Mean Absolute Error*. A métrica consiste na soma de todos os erros absolutos para cada exemplo de teste dividido pelo número de exemplos, conforme a Equação 4.1. À medida que a distância entre o *valorEstimado* e o *valorReal* aumenta, o erro aumenta de forma linear. A mediana dos erros absolutos também foi usada como métrica de avaliação. São considerados todos os erros absolutos para cada exemplo, no entanto, é retornado o valor central dessa lista de erros ordenados. A mediana é mais robusta para avaliar um distribuição com *outliers* ou que não pertence a uma distribuição normal.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |valorEstimado - valorReal| \tag{4.1}$$

2. Desvio Padrão: quando apresentada a média dos erros absolutos, o desvio padrão também foi medido conforme a Equação 4.2, em que x_i indica o erro i e $\bar{x}$ a média dos erros na distribuição, enquanto n é o tamanho do conjunto.

$$\mathrm{DP} = \sqrt{\sum \frac{(x_i - \bar{x})^2}{n}} \tag{4.2}$$

3. Logaritmo do erro absoluto: na Equação 4.3, é apresentado o *Absolute Error* (AE) em escala logarítmica.

$$\mathrm{AE}_{\mathrm{LOG}} = log_{10}^{|valorestimado_i - valorReal_i|}. \tag{4.3}$$

4. Média da magnitude do erro absoluto ajustada: o *Magnitude Relative Error* (MRE) poderia ter sido mais uma estratégia usada para avaliar os modelos, mas (SHEPPERD; MACDONELL, 2012) apresentou problemas com o uso da métrica. Foi apresentado um exemplo de dois projetos de *software*, em que o primeiro teve o valor da variável dependente superestimado e o segundo subestimado. Ambos tinham o mesmo AE, mas os valores do MRE diferiram no grau de importância.

   Consequentemente, os pesquisadores deduziram que o MRE poderia enviesar estimativas em sistemas de predição, pois métodos com estimativas abaixo do valor esperado poderiam ser favorecidos. Os pesquisadores nomearam este problema como *over-optimism*, porém, o MRE é a métrica de avaliação mais utilizada em problemas EES (IDRI; HOSNI; ABRAN, 2016a).

   Desta maneira, a fim de resolver esse impasse, é apresentada uma métrica que soluciona o problema mostrado por (SHEPPERD; MACDONELL, 2012), a qual vamos chamá-la de *Magnitude Relative Error - Adjusted* (MRE-*ADJUSTED*). A equação 4.4 apresenta a fórmula usada para calcular o erro utilizado pelo MRE-*ADJUSTED*.

   Na Tabela 4 é apresentado um exemplo mais robusto do que o exibido em (SHEPPERD; MACDONELL, 2012). Nele, foi usado o resultado de três métodos, são eles: a média das saídas de um conjunto de modelos, além do menor e do maior valor desse conjunto. Para cada método, foram usadas a média do erro absoluto e as medidas do MRE e do MRE-*ADJUSTED*.

   A partir dos resultados apresentados na Tabela 4, podemos perceber que os valores do *Mean Magnitude Relative Error* (MMRE) diferem em larga escala do MAE, enquanto a média do MRE-*ADJUSTED* manteve a proporção do MAE. Perceba que o método MÍNIMO, que estima com a menor saída emitida pelos modelos do CB, foi indicado pelo MMRE como o melhor método, mas de fato ele não foi. O MMRE super valorizou a saída do MÍNIMO porque, conforme apresentado em (SHEPPERD; MACDONELL, 2012), a métrica sofreu de *over-optimism*. A Figura 21 apresenta a matriz de correlação entre os dados de erros da MÉDIA, MÁXIMO e MÍNIMO em relação às métricas. A correlação da MAE com o MRE-*ADJUSTED* foi maior do que a correlação da MAE com o MRE, quanto à avaliação da MÉDIA. A correlação usando o MÁXIMO foi semelhante, mas quando houve a avaliação

do método MÍNIMO, o problema ficou mais aparente, pois a correlação, apesar de pequena, foi negativa. Considerando o método MÍNIMO, a avaliação usando o MMRE tende a indicar que o método obteve boas estimativas. As principais comparações foram marcadas com uma borda ao redor dos valores.

$$\text{MRE}_{\text{ADJUSTED}} = \frac{\frac{|valorEstimado_i - valorReal_i|}{valorEstimado_i} + \frac{|valorEstimado_i - valorReal_i|}{valorReal_i}}{2}. \tag{4.4}$$

Tabela 4 – Avaliação da métrica proposta MRE-*ADJUSTED*

MAE			MMRE-AJUSTADO			MMRE		
MÉDIA	MÁXIMO	MÍNIMO	MÉDIA	MÁXIMO	MÍNIMO	MÉDIA	MÁXIMO	MÍNIMO
2445	2688	2683	**1,97**	2,49	2,77	1,52	2,15	**1,19**

Fonte: O autor (2022)

Figura 21 – Matriz de correlação entre as métricas MAE, MRE-ADJ e MRE

Matriz de correlação entre MAE, MRE-ADJ e MRE

Correlation Matrix

		Média(MAE)	Máximo (MAE)	Mínimo (MAE)
Média(MAE)	Pearson's r	—		
	p-value	—		
Máximo (MAE)	Pearson's r	0.906	—	
	p-value	< .001	—	
Mínimo (MAE)	Pearson's r	0.862	0.724	—
	p-value	< .001	< .001	—
Média (MRE-ADJ)	Pearson's r	0.237	0.186	0.150
	p-value	0.097	0.196	0.299
Máximo (MRE-ADJ)	Pearson's r	0.019	0.024	0.032
	p-value	0.896	0.871	0.827
Mínimo (MRE-ADJ)	Pearson's r	0.052	0.163	0.121
	p-value	0.721	0.258	0.402
Média (MRE)	Pearson's r	0.125	0.109	0.037
	p-value	0.386	0.450	0.799
Máximo (MRE)	Pearson's r	0.015	0.026	0.045
	p-value	0.918	0.857	0.756
Mínimo (MRE)	Pearson's r	0.084	0.065	−0.036
	p-value	0.563	0.656	0.804

Fonte: O autor (2022)

5. Ganho relativo: a fim de verificar quanto cada métrica foi superior a um método simples de *baseline*, foi usado o ganho relativo em relação ao ZEROR (WAIKATO, 2022g), conforme a Equação 4.5.

$$\mathrm{GR} = \frac{(\frac{1}{n}\sum_{i=1}^{n}|estimativaZeroR - valorReal|) - (\frac{1}{n}\sum_{i=1}^{n}|valorEstimado - valorReal|)}{\frac{1}{n}\sum_{i=1}^{n}|estimativaZeroR - valorReal|)} \tag{4.5}$$

6. A performance de cada método durante a criação da τ_c foi medida pelo AE apresentado na Equação 4.6, dado por:

$$\mathrm{AE} = |valorEstimado_i - valorReal_i|. \tag{4.6}$$

em que $valorEstimado_i$ e $valorReal_i$ representam, respectivamente, o valor previsto e a saída desejada para *iésimo* padrão.

Os testes estatísticos são comumente utilizados para verificar se a hipótese que está sendo avaliada é verdadeira, ou melhor, se existem diferenças significativas entre os tratamentos (métodos). Na comparação de diferentes métodos, é importante definir quais superam ou se diferenciam dos demais. O AE foi a métrica usada nos testes estatísticos, conforme recomendado por (SHEPPERD; MACDONELL, 2012).

Um teste estatístico é escolhido basicamente a partir das respostas de três perguntas: (i) os grupos de amostras condizem com uma distribuição normal? (ii) as amostras são dependentes ou independentes? e (iii) qual a quantidade de grupos amostrais? De maneira geral, os testes paramétricos são aplicados em amostras cujos valores propendem a uma distribuição normal, ou seja, a sua distribuição encaminha a ser simétrica. Testes de normalidades podem ser realizados para verificar a presença de normalidade na distribuição.

Os testes não paramétricos são adequados para amostras que não apresentam tendência a uma normalidade. No Capítulo 6, é apresentado o resultado do teste de normalidade de *Shapiro-Wilk* (SHAPIRO; WILK, 1965) para cada grupo de amostras. Além disso, para saber o método estatístico ideal para um experimento, é preciso definir se as amostras são dependentes ou independentes. Considerando que todas as instâncias de treinamento, validação e testes permaneceram constantes para todos os métodos avaliados, podemos dizer que as amostras de erros são dependentes. Por fim, a quantidade de grupos de amostras também influencia na escolha do teste adequado. Neste trabalho, o número de grupos de amostras é igual a quantidade de métodos comparados. Neste sentido, os resultados apresentados neste trabalho foram baseados em testes estatísticos adequados para cada contexto.

O teste T para amostras dependentes pode ser aplicado quando se tem dois grupos de amostras com distribuição normal. Mas, conforme recomendado por (DEMSAR, 2006; HOLLANDER; WOLFE; CHICKEN, 2013), ANOVA e *Friedman* são testes usados para detectar diferenças de desempenho entre três ou mais grupos de amostras. A hipótese nula testada é que todos os algoritmos tenham desempenhos iguais e as eventuais diferenças sejam meramente aleatórias, todavia, eles indicam apenas se houve diferença significativa no resultado de maneira geral. Porém, além disso, é preciso saber em quais métodos as diferenças foram verificadas, caso existam. Nesse sentido, após a rejeição da hipótese nula, um teste *post hoc* deve ser realizado para avaliar o desempenho dos algoritmos em pares. Os testes *T-Pareado* (MORETTIN; BUSSAB, 2010), *ANOVA* (MORETTIN; BUSSAB, 2010), *Friedman* (FRIEDMAN, 1940) e o *post-hoc Least Significant Difference* (LSD) (KESELMAN; KESELMAN; GAMES, 1991) foram aplicados dentro dos contextos adequados. Todos os testes estatísticos de comparação múltiplas foram executados em computador pessoal usando os softwares *Matlab R2018* e *Jamovi* com um nível de confiança de 95%.

5 REPOSITÓRIOS DE DADOS

Os testes realizados neste trabalho consideraram a maioria dos banco de dados de EES conhecidos na literatura, conforme (IDRI; HOSNI; ABRAN, 2016a; IDRI; HOSNI; ABRAN, 2016b). Algumas bases foram descartadas porque não alcançaram a quantidade mínima de exemplos estabelecida neste trabalho (25 exemplos). O *International Software Benchmarking Standard Group* (ISBSG) (ABRAN, 2015) e o *Predictor Models In Software Engineering Repository* (PROMISE) (SHIRABAD; MENZIES, 2005) são repositórios de dados de projetos de *software* comumente utilizados dentro do contexto de EES (WEN et al., 2012; IDRI; HOSNI; ABRAN, 2016a). A fim de estender as descobertas e dar mais robustez as respostas das questões de pesquisa, seis conjuntos de dados com índices educacionais também foram investigados. Esses dados procedem de um estudo realizado por Nascimento et al. (NASCIMENTO; FAGUNDES; MACIEL, 2019).

O primeiro repositório analisado procede do ISBSG (ABRAN, 2015). Este repositório contém dados oriundos de diferentes organizações ao redor do mundo e, por isso, o repositório é caracterizado como heterogêneo. A primeira análise considerou 1466 exemplos do ISBSG e o segundo repositório analisado foi o PROMISE (SHIRABAD; MENZIES, 2005), o qual possui várias bases de dados de engenharia de *software*. Os dados das bases do PROMISE são mais homogêneos do que os do ISBSG, visto que cada base de dados investigada vem da mesma organização mas, em contrapartida, as bases de dados do PROMISE são menores do que a base do ISBSG. Para a avaliação dos resultados deste repositório, cada base de dados foi testada separadamente, no entanto, para simplificar a análise, os métodos foram comparados por repositório, sendo que, no total, elas somaram 898 exemplos. Por fim, temos os dados educacionais que, semelhante ao PROMISE, foram testados separadamente. Ou seja, o processo de treinamento, validação e testes foi realizado individualmente para cada conjunto de dados, todavia, a análise principal dos resultados foi realizada a partir dos resultados do repositório de forma geral. As bases de dados educacionais somaram 21.022 exemplos.

A Tabela 5 apresenta as características de cada repositório e das respectivas bases de dados. A coluna Fonte indica se os dados são oriundos de uma mesma organização, enquanto, as demais colunas, informam a quantidade total de atributos, a quantidade de atributos categóricos, de atributos numéricos e, por fim, o tamanho da base de dados. Nas subseções seguintes é apresentada uma visão geral dos repositórios.

Tabela 5 – Características dos repositórios e bases de dados usados no experimento

Repositório	Base de dados	Fonte	Atrib.	Cat.	Num.	Instân.
ISBSG	ISBSG - All	Variadas	16	12	3	1466
PROMISE	China	Única	14	0	13	346
PROMISE	Cocomo81	Única	18	1	16	63
PROMISE	Cocomonasa V1	Única	17	15	1	60
PROMISE	Cocomonasa V2	Única	23	20	2	93
PROMISE	Desharnais	Única	11	1	9	81
PROMISE	Kitchenham	Única	6	3	2	145
PROMISE	Maxwell	Única	27	0	26	62
PROMISE	Miyazaki94	Única	8	0	7	48
EDUCACIONAL	Ap. En. Fun.	Única	10	0	9	8132
EDUCACIONAL	Ap. En. Méd.	Única	10	0	9	1135
EDUCACIONAL	Ev. En. Fun.	Única	9	0	8	3683
EDUCACIONAL	Ev. En. Méd.	Única	10	0	9	468
EDUCACIONAL	Re. En. Fun.	Única	10	0	9	6592
EDUCACIONAL	Re. En. Méd.	Única	10	0	9	1012

Fonte: O autor (2022)

5.1 ISBSG

O ISBSG contém um repositório de dados de desenvolvimento e melhoria de software, além de um repositório de manutenção e suporte. Nesse sentido, algumas informações são importantes quanto aos dados do repositório atual, cuja versão mais recente consiste em mais de 10.000 projetos de *softwares*, os quais foram desenvolvidos em 32 países e em uma ampla variedade de organizações. Porém, os dados utilizados nesse trabalho foram extraídos do *Release* 11, onde o repositório possuía pouco mais de 5.000 projetos de *softwares* oriundos de 25 países.

É sabido que a construção e a avaliação de técnicas de EES contam principalmente com: (i) dados históricos de projetos de *software*, e (ii) métodos com ou sem aprendizagem (IDRI; HOSNI; ABRAN, 2016a; WEN et al., 2012). Portanto, a acurácia das técnicas de estimativas depende das características dos dados, tais como, tamanho, valores ausentes, *outliers*, quantidade de atributos, tipos de dados etc. Além desses, a escolha dos métodos usados para avaliação também contribue para a variação dos resultados. Os dados do ISBSG foram usados, a fim de avaliar os métodos do estado da arte e o *framework* proposto, visto que eles são relevantes na literatura (IDRI; HOSNI; ABRAN, 2016a) e possuem características distintas daquelas encontradas nas bases de dados do PROMISE.

Considerando que diferentes valores variam simultaneamente, os efeitos estatísticos podem ser mais difíceis de serem identificados em banco de dados heterogêneos do que

nos homogêneos (GENCEL; BUGLIONE, 2008). As bases heterogêneas são normalmente construídas a partir de dados oriundos de diferentes organizações, os quais são inseridos no repositório sem as suas identificações exatas, quanto à organização de origem, enquanto nas homogêneas, existem mais semelhanças quanto à origem dos dados. De maneira geral, os valores das variáveis variam menos.

Diante do que já foi dito, quanto à fonte dos dados, a seguir, será apresentada a extração que foi aplicada ao conjunto de dados do repositório do ISBSG *Release* 11. Os dados foram filtrados com as seguintes características:

- A taxa de qualidade dos dados extraídos igual a: A ou B.

  Projetos com qualidade inferior não são recomendados. Essa taxa é descrita pelo repositório do ISBSG, em que todos os projetos são categorizados com valores de A a D.

- Linguagens de programação primária: *Java*, *C++* ou *Visual Basic*.

  No momento da extração essas eram as linguagens de programação com maior quantidade de dados.

- Dimensão de tamanho dos projetos: pontos de função.

  A intenção de escolher um único tipo de dimensão foi para evitar o uso de diferentes medidas, uma vez que o tamanho do projeto influencia no esforço. Isto levaria a ter que transformar os valores entre as medidas.

A partir da filtragem aplicada no repositório, 1466 projetos com 38 características foram extraídos da base de dados do ISBSG. No entanto, os dados do ISBSG contêm algumas particularidades. Por exemplo, por serem privados, cada pesquisador pode extrair exemplos distintos do repositório, e normalmente as extrações dos dados do ISBSG não trazem um conteúdo pronto para uso em algoritmos de Aprendizagem de Máquina (AM). Nesse sentido, os dados foram pré-processados, pequenos ajustes foram necessários, a fim de melhorar a qualidade dos dados extraídos, entre as mudanças, destaca-se:

1. Eliminação manual de atributos não relevantes, por exemplo, atributos textuais que descrevem o atributo de entrada;
2. Eliminação de atributos não referentes ao desenvolvimento de *software*;
3. Eliminação de atributos com informação duplicada;
4. Imputação de dados ausentes. Atributos categóricos com dados ausentes receberam uma nova categoria de informação;
5. Imputação de dados ausentes contínuos. Foi usado a média ou, quando adequado, o preenchimento manual com um valor constante.

Conforme mencionado acima, foi necessária a eliminação manual de alguns atributos de entrada. Esses atributos foram eliminados baseando-se em uma análise individual quanto aos valores e influência do atributo na predição. São exemplos de atributos não relevantes e que foram eliminados: (i) a identificação do projeto, que é único para cada exemplo; (ii) o nível de qualidade da coleta dos dados, que se refere a uma informação interna do repositório; (iii) o tempo do projeto em meses, que entraria em conflito com a variável dependente esforço, além de conter muitos dados ausentes; (iv) atributos redundantes, que trazem informações já encontradas em outros atributos; e finalmente, (v) atributos que trazem informações descritivas sem relevância para o domínio do problema, como, por exemplo, o versionamento de ferramentas usadas nos projetos. O conjunto de dados do ISBSG não apresentou instâncias repetidas.

Após a análise individual dos atributos, 21 atributos de entrada foram eliminados, além do atributo *summary work effort* que registra o esforço do projeto sem considerar o ciclo de desenvolvimento completo. Projetos que não foram registrados com o esforço de, pelo menos, um ciclo de desenvolvimento completo, são estimados a partir do esforço registrado no *summary work effort* e das fases restantes para o ciclo de vida se completar. O atributo *normalised work effort* apresenta esses valores de maneira mais consistentes e foi escolhido como variável dependente, conforme sugerido por (GUEVARA; FERNáNDEZ-DIEGO; LOKAN, 2016).

Dos 15 atributos de entrada que restaram, 12 continham valores nulos e, dessa forma, foi necessário imputar valores nesses dados ausentes. No entanto, a maioria dos atributos com dados ausentes não permitiam uma imputação manual precisa, uma vez que, preencher dados ausentes manualmente com alta precisão a partir dos 15 atributos e de 1466 exemplos, não é razoavelmente adequado. Neste sentido, as estratégias tomadas para imputar dados categóricos e numéricos foram:

1. Dados categóricos: um novo valor representando uma nova categoria para o atributo com dados ausentes. Essa estratégia foi usada para valores categóricos nominais, cuja base de dados não contém valores ordinais.

2. Dados numéricos: a média dos valores, ou valores constantes coerentes com a variável, como é o caso dos valores 0 e 1.

A solução adotada de criar um novo valor para atributos com dados ausentes é apresentada como uma das possíveis técnicas de imputação de dados em (FACELI et al., 2011). No entanto, para garantir que a estratégia possa melhorar a qualidade dos dados, testes estatísticos foram realizados e uma análise usando o teste de *Kruskal Wallis* foi aplicada.

O teste de Kruskal-Wallis (KRUSKAL; WALLIS, 1952) é não paramétrico e utilizado para comparar três ou mais populações. O método investiga a hipótese de que todas as amostras possuam funções de distribuições iguais, contra a hipótese alternativa de que pelo menos duas das amostras possuam funções de distribuições diferentes. O teste é

indicado quando: (i) as amostras não tendem a uma distribuição normal, (ii) as amostras são independentes, e (iii) tem-se mais do que 2 amostras a serem testadas. Todos esses requisitos foram obedecidos pelas amostras avaliadas.

Entretanto, o teste de *Kruskal-Wallis* não informa quais grupos podem ou não ser distintos. Isso leva à necessidade de utilizar um procedimento de comparações múltiplas para determinar quais os grupos que possuem distribuições significativamente diferentes, já que a literatura aborda diferentes testes de comparações múltiplas (HSU, 1996), os quais podem ser utilizados juntamente com o teste de *Kruskal-Wallis*. Neste trabalho, para o contexto abordado, foi usado o teste de *Bonferroni*.

Todos os atributos de entrada categóricos que continham dados ausentes foram avaliados com o teste de *Kruskal-Wallis* e posteriormente com o teste de *Bonferroni*. Os resultados mostraram que existiam diferenças significativas ($\alpha = 0{,}05$) entre o novo valor criado no atributo e ao menos uma das categorias nominais existentes. Após essas análises, o conjunto de dados extraído do repositório do ISBSG e usado nos experimentos deste trabalho continuou contendo 1466 instâncias, porém, sem dados ausentes distribuídos entre os 15 atributos.

Entre os atributos de entrada do ISBSG, 12 são categóricos e 3 numéricos, porém, alguns algoritmos de AM são limitados à manipulação de valores não numéricos. Desta forma, é razoável converter esses valores categóricos para numéricos, de maneira que seja possível utilizar os modelos com essa limitação. Sempre que necessário, os valores categóricos foram transformados em variáveis *Dummies* (SUITS, 1957), e os valores numéricos foram reescalados entre 0 e 1.

Ao fim do pré-processamento, restaram 15 variáveis independentes e 1 dependente, que é detalhada na Tabela 6 e está distribuída conforme as Figuras 22 e 23. É perceptível que a variável tem uma distribuição assimétrica à direita e, consequentemente, os dados não se tornam propensos a uma distribuição normal. Essa informação foi confirmada com a aplicação do teste de *Shapiro-wilk* (SHAPIRO; WILK, 1965), que foi aplicado aos dados e o resultado apresentou o valor de *p-value* abaixo de 0,0001%. O valor da estatística do teste é apresentado na última coluna da Tabela 6.

A variável dependente se relaciona com as variáveis independentes numéricas de acordo com a Figura 24. Nos gráficos, é possível perceber uma correlação forte das variáveis *adjusted function points* e *max team size* com a variável dependente, o que já era de se esperar, visto que, quanto maior um projeto de *software*, maior tende a ser o esforço a desenvolvê-lo e, de maneira semelhante, quanto maior o tamanho da equipe, maior tende a ser o projeto. O atributo *quantity documents* não obteve alta correlação com o esforço do projeto, o que também já era esperado, uma vez que o número de artefatos documentais gerado tende a estar mais relacionado com o tipo de metodologia de desenvolvimento do que com o esforço de um projeto de *software*. No entanto, o atributo continuou na base porque ele poderá contribuir na predição dos modelos. Por estar fora do escopo deste

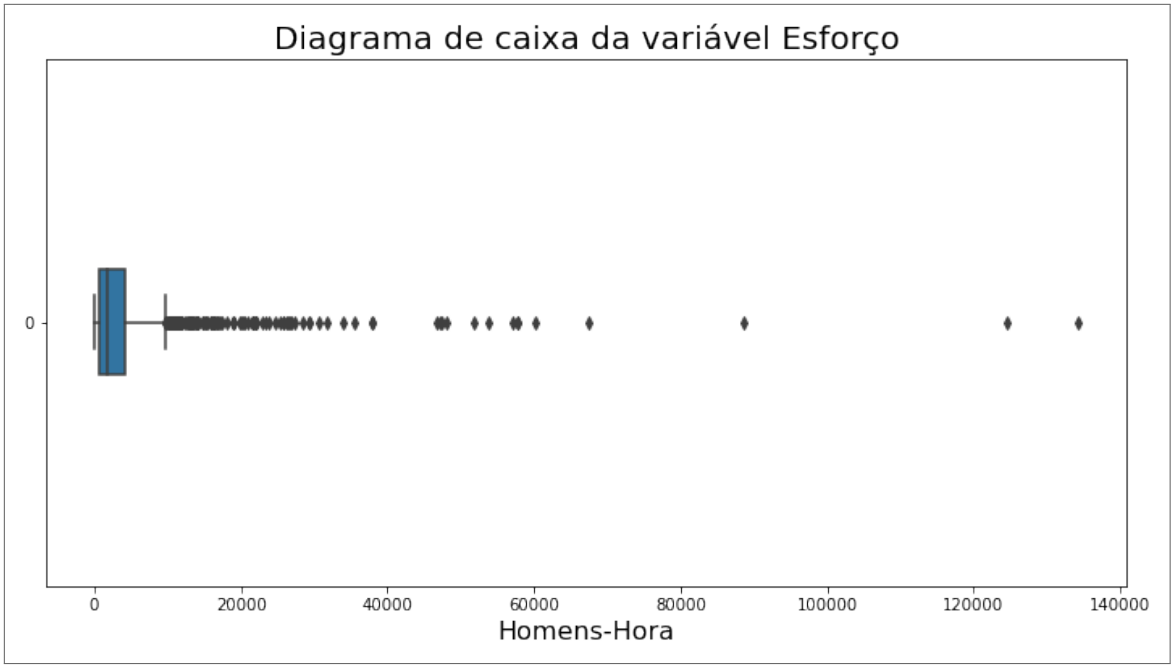
trabalho, métodos de seleção de atributos não foram previamente aplicados nos conjuntos de dados, de maneira que os atributos que permaneceram no conjunto de dados após o processo de eliminação manual foram as variáveis preditoras utilizadas para as previsões da variável dependente.

Tabela 6 – Informações da variável dependente da base de dados do ISBSG

Base de dados	Mín.	Máx.	Média	Med.	Des.Pad.	Coe.Var.	Ass.	Cur.	Nor.
ISBSG	8	134211	4190,31	1844	8346,56	1,99	7,55	86,06	0,43

Fonte: O autor (2022)

Figura 22 – Distribuição dos valores da variável dependente do ISBSG no gráfico *box-plot*.

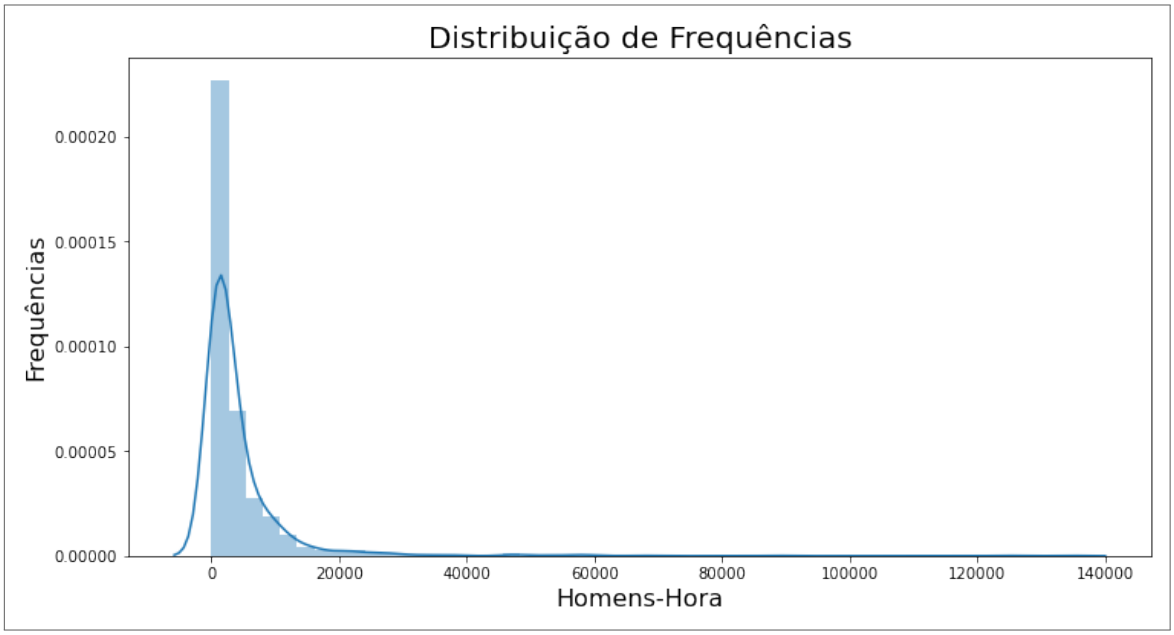


Fonte: O autor (2022)

Diante do que foi apresentado, pode-se perceber que:

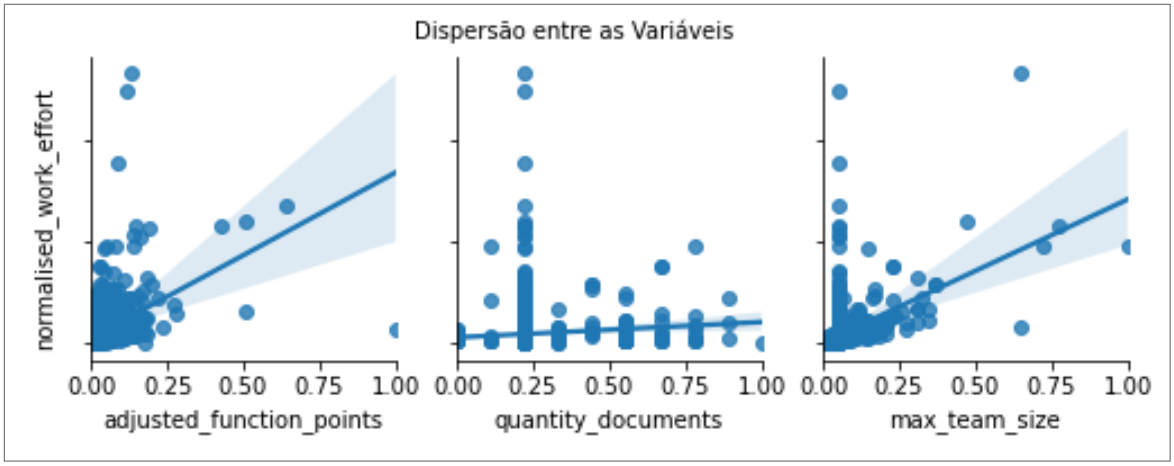
- Os dados da variável dependente do ISBSG não tendem a uma distribuição normal, de acordo com a distribuição dos dados;
- A distribuição é assimétrica à direita, visto o valor de assimetria na Tabela 6 e a Figura 23;
- Dos 15 atributos de entrada, 12 são nominais e 3 numéricos;
- O desvio padrão é maior do que a média, o que indica uma alta variação dos dados em relação ao valor médio, e é confirmado pelo valor do coeficiente de variação;
- Das variáveis numéricas, duas apresentam tendência a ter correlação com a variável dependente;

Figura 23 – Distribuição de frequência da variável dependente Esforço em Homens-Hora do ISBSG .



Fonte: O autor (2022)

Figura 24 – Dispersão das variáveis independentes numéricas com a variável dependente Esforço.



Fonte: O autor (2022)

- Em relação ao achatamento da curva de distribuição dos dados, apesar de não estarmos tratando de uma distribuição normal, é perceptível uma distribuição mais estreita em torno da assimetria;
- A variável dependente, aparentemente, contém valores discrepantes, mas que foram analisados e, de fato, são reais e permaneceram no conjunto de dados.

Destacamos essas características para que possam ser analisadas junto aos resultados apresentados no Capítulo 6. A Figura 48 do Apêndice A apresenta a distribuição dos dados de cada atributo do ISBSG. As variáveis em branco referem-se aos valores numéricos, enquanto as em preto aos nominais.

5.2 PROMISE

O repositório do PROMISE contém bases de dados com características mais homogêneas com relação à coleta do que os do ISBSG. Enquanto os dados do ISBSG contêm características que levam à necessidade de um pré-processamento mínimo, os dados oriundos do PROMISE basicamente: (i) não contém dados ausentes, exceto no conjunto de dados *Kitchenham*, onde o atributo *ProjectType* continha 10 exemplos ausentes, mas que foram preenchidos com uma nova categoria; (ii) não possuem instâncias repetidas nas bases (iii) são coletados a partir da mesma origem; (iv) as bases de dados são menores e (v) são públicos e disponíveis gratuitamente.

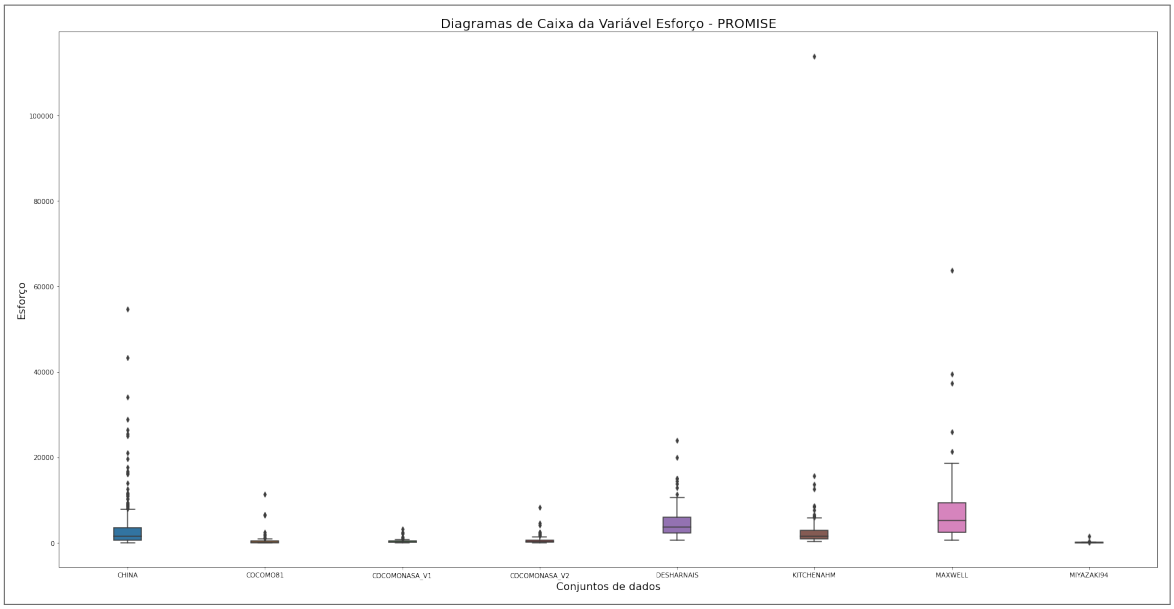
Entre as variáveis independentes das bases de dados do PROMISE, tem-se atributos categóricos e numéricos. No entanto, semelhante ao que foi realizado nos dados do ISBSG, as variáveis categóricas foram convertidas em valores numéricos para os algoritmos de AM, que são limitados à manipulação de valores numéricos. Sempre que necessário, os valores nominais foram transformados em variáveis *Dummies* (SUITS, 1957), e os valores numéricos foram reescalados entre 0 e 1.

A fim de conhecer mais sobre a distribuição das variáveis dependentes dos oito conjuntos de dados do PROMISE, são apresentadas na Tabela 7, as estatísticas descritivas de cada uma das variáveis. As Figuras 25 e 26 disponibilizam o diagrama de caixa e a distribuição de frequência, respectivamente, das variáveis dependentes. Com o intuito de comparar os valores das variáveis dependentes e permitir melhor visualização dos dados, a Figura 27 apresenta o logaritmo dos valores originais dessas variáveis.

Os resultados dessas estatísticas mostraram que os dados do PROMISE não se inclinam a uma distribuição normal e, com o fim de confirmar essa afirmação, o teste de *shapiro-wilk* (SHAPIRO; WILK, 1965) foi aplicado. O resultado deste teste apresentou o *p-value* abaixo de 0,0001%, o que indica uma forte tendência aos dados não possuírem uma distribuição normal. As estatísticas dos testes de normalidade de *Shapiro-Wilk* são apresentadas na última coluna da Tabela 7.

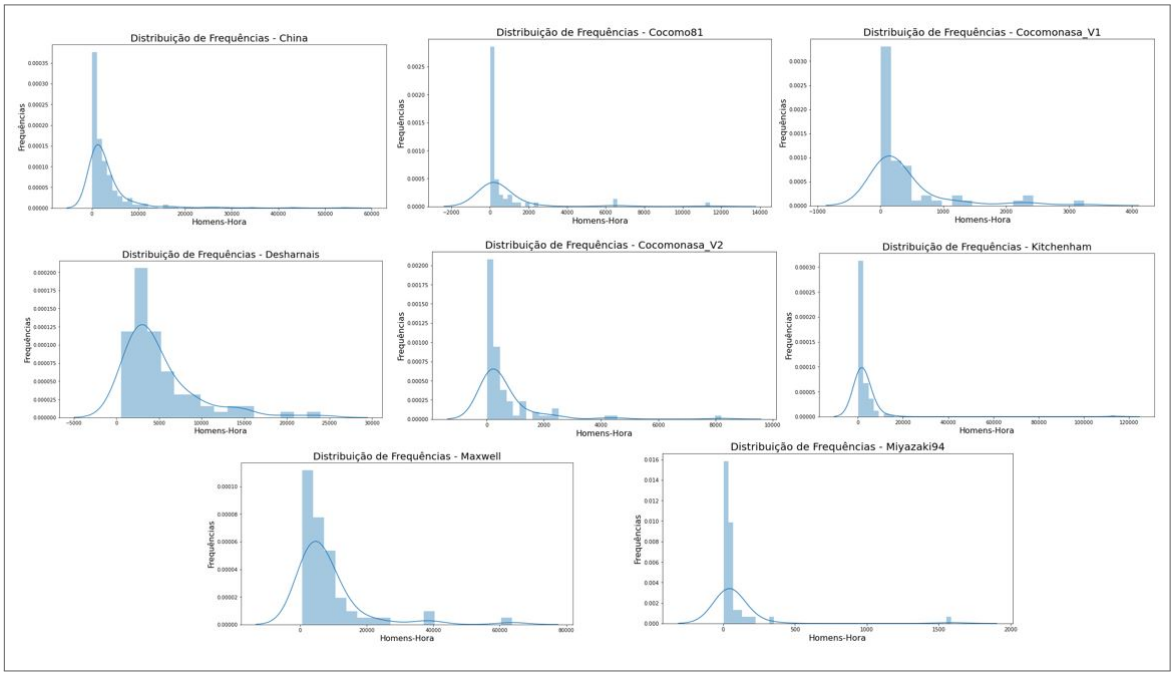
Diante do que foi apresentado, pode-se perceber que:

Figura 25 – Diagramas de caixa das variáveis dependentes das bases de dados do PROMISE



Fonte: O autor (2022)

Figura 26 – Diagramas de frequência das variáveis dependentes do PROMISE



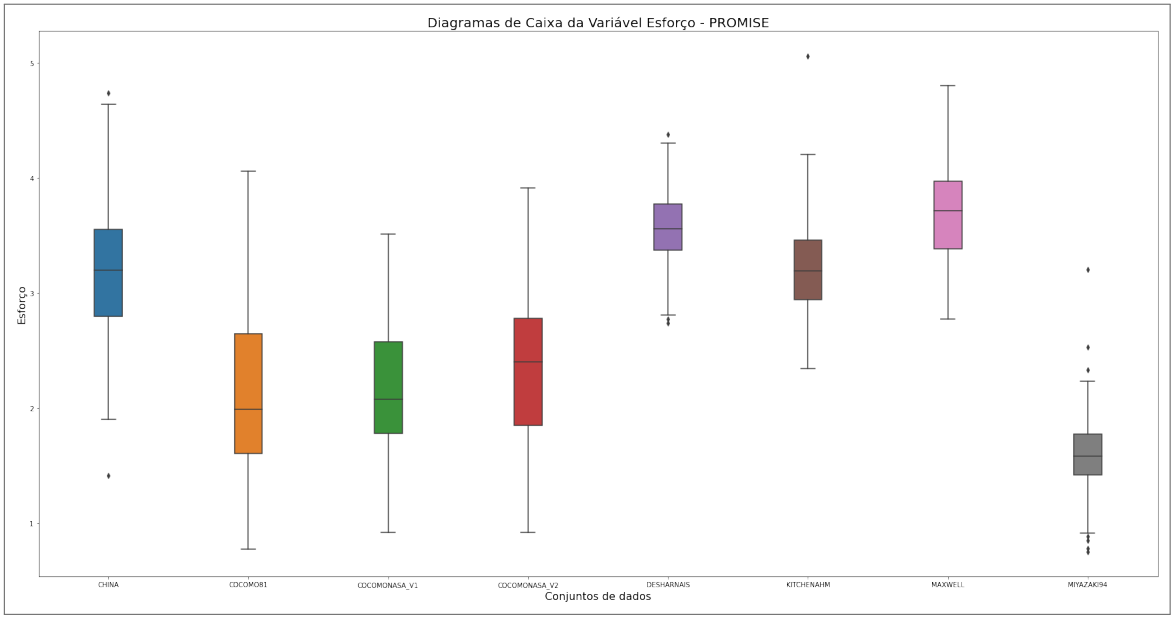
Fonte: O autor (2022)

Tabela 7 – Informações das variáveis dependentes do PROMISE

Base de dados	Mín.	Máx.	Média	Med.	Des. Pad.	Coe.Var.	Ass.	Cur.	Nor.
China	26	54620	3360,01	1568	5715,07	1,7	4,69	29,23	0,52
Cocomo81	5,9	11400	683,53	98	1821,51	2,66	4,26	19,35	0,39
Coc. NasaV1	8,4	3240	406,41	118,80	656,97	1,62	2,55	6,31	0,61
Coc. NasaV2	8,4	8211	624,41	252	1135,93	1,82	4,12	21,28	0,53
Desharnais	546	23940	5046,31	3647	4418,77	0,88	1,93	4,18	0,8
Kitchenham	219	113 930	3113,12	1557	9598,01	3,08	10,64	119,60	0,19
Maxwell	583	63694	8223,21	5189,50	10499,9	1,28	3,19	12,02	0,63
Miyazaki94	5,6	1586	87,48	38,1	228,76	2,62	5,88	35,40	0,29

Fonte: O autor (2022)

Figura 27 – Diagramas de caixa do logaritmo das variáveis dependentes das bases de dados do PROMISE



Fonte: O autor (2022)

- Semelhante ao ISBSG, os dados das variáveis dependentes do PROMISE não se conduzem a uma distribuição normal;
- Todas as distribuições tendem a ser assimétricas à direita;
- Em geral, as bases de dados contêm mais atributos numéricos do que categóricos;
- Exceto para o *Desharnais*, todos os valores de desvio padrão são maiores do que os da média, o que leva a termos dados mais variados;
- É perceptível em dois grupos, de bases de dados em relação aos valores da variável dependente, um com valores predominantemente mais altos e outro com valores mais baixos;

- A variabilidade dos dados também é um caracterizador da variável dependente, já que 3, das 8 bases de dados, se destacam-se quanto ao alto valor do coeficiente de variação, o que indica valores mais dispersos e provavelmente com a presença de dados discrepantes.

As características encontradas nas variáveis dependentes das bases de dados do PROMISE serão analisadas no Capítulo 6 quanto à existência de relação dessas variáveis com os resultados apresentados.

Todas as distribuições dos conjuntos de dados do PROMISE são apresentadas no Apêndice A, nas Figuras 49, 50, 51, 52, 53, 54, 55 e 56. As variáveis em branco referem-se aos valores numéricos, enquanto as em preto referem-se aos nominais.

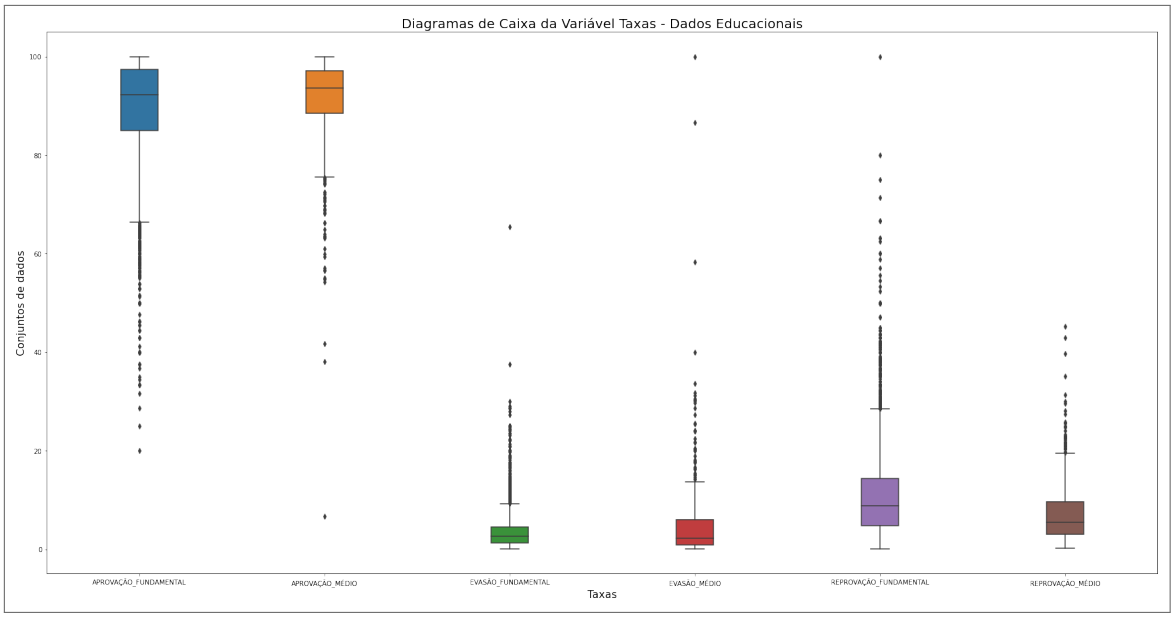
5.3 DADOS EDUCACIONAIS

O estudo de AM em dados educacionais está relacionado ao desenvolvimento e à pesquisa de métodos que são aplicados, a fim de encontrar padrões nos dados coletados. Desta maneira, é possível descobrir e abordar novos fenômenos dentro de algum cenário educacional. Os dados usados nesta pesquisa são indicadores educacionais em escolas do Brasil de nível fundamental e médio. A *Education Data Mining* (EDM) pode abordar desde análises estatísticas descritivas até a aplicação de métodos de AM. As revisões (ROMERO; VENTURA, 2010a; ROMERO; VENTURA, 2013) apresentam os estudos mais relevantes realizados neste campo de pesquisa.

A coleta dos dados usados nessa terceira abordagem de testes foi realizada pelos autores do estudo (NASCIMENTO; FAGUNDES; MACIEL, 2019) a partir de um processo baseado no CRISP-DM (CHAPMAN et al., 2000), e dos dados disponíveis pelo Instituto Nacional de Educação e Pesquisa (INEP)(AZEVEDO, 2018). Os indicadores usados nos conjuntos de dados foram referentes ao ano de 2016 e as taxas de eficiência escolar avaliadas foram divididas em seis cenários: (i) a taxa de aprovação no ensino fundamental; (ii) a taxa de aprovação no ensino médio; (iii) a taxa de evasão no ensino fundamental; (iv) a taxa de evasão no ensino médio; (v) a taxa de reprovação no ensino fundamental e (vi) a taxa de reprovação no ensino médio. Os dados coletados possuem características homogêneas, visto que são oriundos das mesmas fontes. Além disso, eles apresentam pequenas variações. A Tabela 5 apresenta informações sobre os conjuntos de dados, enquanto a Tabela 8 sumariza as características da variável dependente.

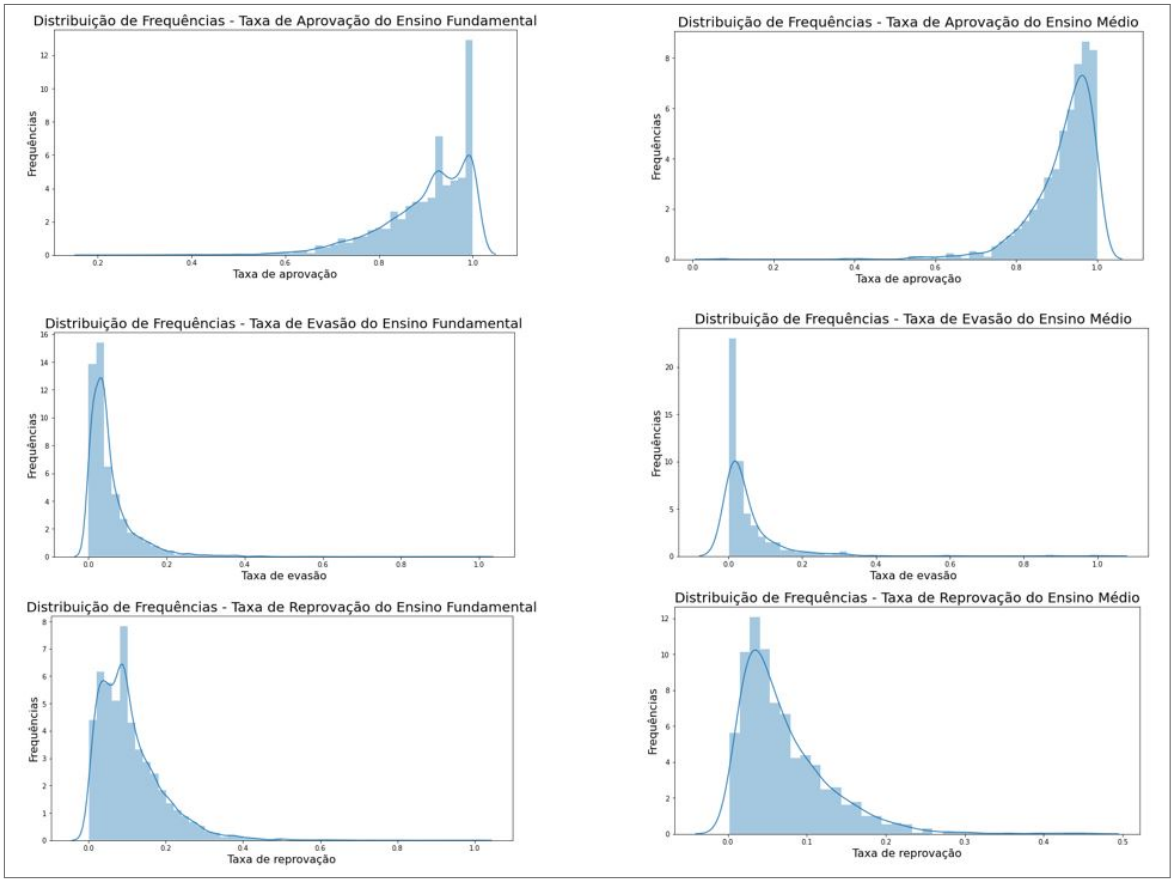
Uma característica importante que podemos perceber nas variáveis dependentes dos dados educacionais é a proximidade entre os valores da média e mediana, conforme apresentado na Tabela 8, enquanto que isso não acontece nas bases de dados dos repositórios anteriores. Este fato poderia caracterizar uma distribuição normal na variável dependente, no entanto, de acordo com o diagrama de caixas apresentado na Figura 28 e as distribui-

Figura 28 – Diagramas de caixas das variáveis dependentes dos dados educacionais



Fonte: O autor (2022)

Figura 29 – Diagramas de frequência das variáveis dependentes dos dados educacionais



Fonte: O autor (2022)

Tabela 8 – Informações das variáveis dependentes dos dados educacionais

Base de dados	Mín.	Máx.	Média	Med.	Des Pad.	Coe.Var.	Ass.	Cur.	Nor.
Apr. Ens. Fun.	20	100	89,86	92,20	9,71	0,11	-1,49	3,37	0,90
Apr. Ens. Méd	6,7	100	91,57	93,60	8,24	0,09	-2,66	14,4	0,80
Eva. Ens. Fun.	0,1	65,5	3,68	2,6	3,94	1,07	3,46	24,88	0,71
Eva. Ens. Méd.	0,1	100	5,25	2,3	8,94	1,7	5,38	43,45	0,53
Rep. Ens. Fun.	0,1	100	10,72	8,90	8,36	0,78	1,90	7,24	0,87
Rep. Ens. Méd.	0,20	45,20	7,13	5,5	5,68	0,8	1,86	5,71	0,85

Fonte: O autor (2022)

ções de frequências apresentada na Figura 29, podemos perceber que as distribuições das variáveis dependentes são assimétricas.

A fim de investigar a existência de normalidade na distribuição dos dados das variáveis dependentes, foi realizado o teste de normalidade de *Shapiro-Wilk* (SHAPIRO; WILK, 1965). Os resultados apresentaram os valores de $p - value$ abaixo de 0,0001%, o que indica que a distribuição não tende a uma normalidade. O resultado da estatística do teste também foi apresentado na última coluna da Tabela 8. Todas essas informações serão consideradas na análise dos resultados apresentada no Capítulo 6.

Enquanto as taxas apresentadas na Tabela 8 referem-se às variáveis dependentes, 17 variáveis preditoras foram analisadas quanto à sua inferência na variável dependente para cada conjuntos de dados educacionais. No estudo feito por (NASCIMENTO; FAGUNDES; MACIEL, 2019), os autores avaliaram a seleção de atributos usando correlação e *Random Forest* e, de acordo com os resultados do estudo, o uso do *Random Forest* para seleção de atributos obteve melhores resultados do que a correlação. Nesse sentido, foram considerados os cenários com as variáveis independentes selecionadas pelo *Random Forest* e que foram identificadas em (NASCIMENTO; FAGUNDES; MACIEL, 2019).

Diante do que foi apresentado, pode-se perceber que:

- Os dados das variáveis dependentes dos dados educacionais não tendem a uma distribuição normal;
- Em relação a distribuição dos dados, 4 das 6 bases possuem distribuição assimétrica à direita, e 2 à esquerda, conforme Figura 29;
- O desvio padrão é menor do que a média, o que indica menor variação dos dados em relação ao valor médio, quando comparado com os dados anteriores;
- Em relação ao achatamento da curva de distribuição dos dados, podemos perceber que os dados dos conjuntos de evasão são mais estreitos do que os de aprovação e reprovação;

- A variável dependente aparentemente contém valores discrepantes, mas que, de fato, são reais e permaneceram no conjunto de dados.

Destacamos essas características para que possam ser analisadas junto aos resultados apresentados no Capítulo 6. No Apêndice A, é apresentada nas Figuras 57, 58, 59, 60, 61 e 62 a distribuição dos dados para cada cenário educacional.

6 RESULTADOS

Neste capítulo, serão apresentados os resultados das aplicações dos métodos de estimativas usados nesta tese. A abordagem da apresentação dos resultados foi realizada por repositório e, dessa forma, as Seções 6.1, 6.2 e 6.3 apresentam os resultados das comparações entre os métodos, considerando os repositórios: *International Software Benchmarking Standard Group* (ISBSG); *Predictor Models In Software Engineering Repository* (PROMISE) e os educacionais, respectivamente.

Os testes realizados neste estudo foram baseados em problemas de regressão, portanto, todas as variáveis dependentes possuem valores contínuos. Os tipos e a quantidade de variáveis independentes variam de acordo com os bancos de dados. Nesse sentido, o objetivo dos métodos avaliados é prever o valor da variável dependente, a partir dos dados das variáveis independentes (preditoras). Nesse sentido, cada método de previsão estará associado a um conjunto de erros de estimativas. De maneira geral, diferentes métodos foram analisados neste estudo. Os experimentos realizados têm a finalidade de avaliar o desempenho de cada um deles, a partir do ponto de vista de um analista de dados, dentro do contexto das bases de dados utilizadas.

A fim de definir um desenho experimental, foi definido que cada método avaliado é tido como um nível ou item do fator do experimento (MATOS DANIEL ABUD SEABRA; RODRIGUES, 2019). Neste sentido, a variável resposta estará relacionada a um conjunto de resíduos (erros de estimativas), os quais serão dimensionados de acordo com as métricas apresentadas na Seção 4.4.4. Desta forma, podemos dizer que cada mudança de nível do fator implicará na alternância de um tratamento aplicado ao conjunto de unidades experimentais, ou seja, no contexto dos experimentos, cada método avaliado será identificado como um tratamento distinto.

Quanto às unidades experimentais, temos que, cada instância ou exemplo da base de dados de teste - onde os modelos são avaliados - é tida como uma unidade experimental. Nos testes realizados, as unidades experimentais são randomizadas para cada iteração, no entanto permanecem constantes para todos os tratamentos (métodos). Assim, é possível garantir que as alterações nas variáveis de resposta estarão associadas, apenas, com as mudanças dos tratamentos. De acordo com o que foi abordado até o momento, os experimentos realizados são caracterizados como de um único fator com amostras pareadas. Nas próximas seções serão apresentados os resultados dos testes realizados nos conjuntos de dados usados neste trabalho.

6.1 ISBSG

O Item 6.1.1 aborda os resultados de validação, em seguida é apresentado no Item 6.1.2 os resultados nas bases de testes do ISBSG.

6.1.1 Fase de validação

Na fase de validação, são definidos os Conjunto Básico (CB) e os Conjunto de Seletores (CS) de cada base de dados. A avaliação foi realizada a partir de 1466 instâncias extraídas do ISBSG, de modo que, no geral, para cada iteração, 75% dos dados foram usados para treinamento e validação dos modelos. No total, 72 modelos oriundos de 14 algoritmos de regressão foram avaliados. A fim de obter o tamanho mínimo da amostra para realizar as avaliações, foi realizado o cálculo de tamanho da amostra para um $\alpha = 5\%$, $\beta = 80\%$ e $effectsize = 0,5$, o que resultou em 34 amostras para o teste T-pareado. Nesse sentido, a avaliação dos modelos individuais foi repetida 50 vezes e a média dos erros de cada iteração para cada método foi calculada, baseada nos grupos das amostras.

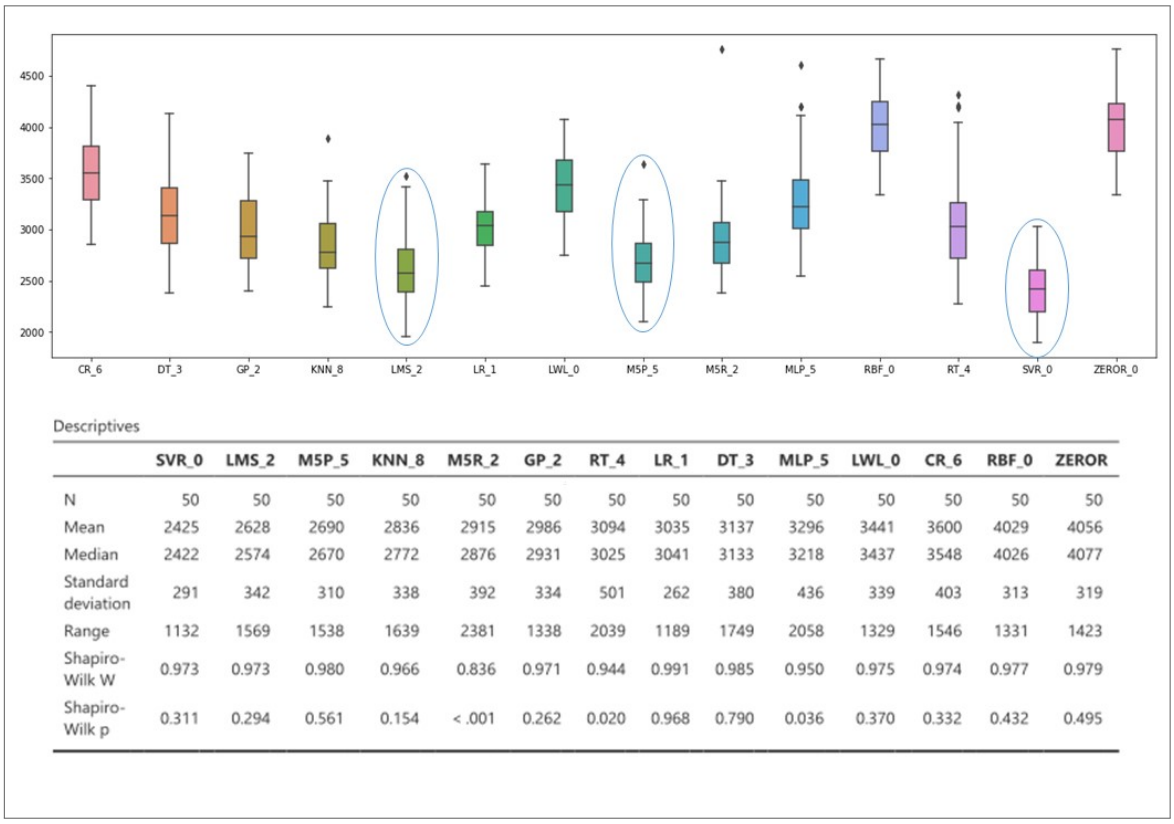
O melhor modelo de cada algoritmo na fase de validação foi selecionado e, em seguida, foram comparados entre si, de maneira que 14 modelos de diferentes algoritmos foram comparados quanto à *Mean Absolute Error* (MAE). A Figura 30 exibe o resultado da distribuição dos erros de cada modelo em forma de gráfico de caixa. Os três melhores modelos foram marcados com um círculo em azul. Ainda na Figura 30 são apresentadas as estatísticas descritivas das amostras de erros de cada modelo avaliado. Com o intuito de facilitar a avaliação, a lista de modelos foi colocada em ordem crescente quanto à média dos erros, concluindo que o SVR_0 foi o melhor modelo avaliado, enquanto o ZEROR foi o pior. O SVR_0 também obteve o menor desvio padrão e a menor amplitude entre os erros. A amostra de erros do M5R_2 obteve baixa probabilidade de ser uma distribuição normal, então, para uma comparação estatística par a par entre os 14 modelos, são indicados os testes de *Friedman* e um teste *poshoc* em seguida, uma vez que o teste de *Friedman* não avalia se existe diferença significativa entre os pares (FRIEDMAN, 1940). O resultado do teste de *Friedman* junto ao teste *poshoc* da *Least Significant Difference* (LSD) é apresentado na Figura 31, e o resultado confirma o que foi apresentado anteriormente na Figura 30.

Para que a avaliação realizada fique mais clara, a Figura 32 apresenta a distribuição dos erros de todos os modelos dos algoritmos vencedores e dos modelos selecionados por cada algoritmo, em: (a) os erros de validação dos modelos do algoritmo *Support Vector Regression* (SVR); (b) do *Least Median Squared* (LMS); (c) do *M5 Base* (M5P); (d) dos modelos selecionados por cada algoritmo, inclusive dos três mostrados em (a), (b) e (c); e em (e) os mesmos valores apresentados em (d), mas no formato de histograma com a densidade das amostras dos erros. A mesma análise apresentada em (a), (b) e (c) na Figura 32 foi realizada nos demais onze algoritmos que não estiveram entre os três melhores, no

entanto, como esses algoritmos não foram selecionados para o CB, não necessitava que os resultados individuais deles fossem apresentados. Os resultados dos melhores modelos selecionados em cada um dos algoritmos são mostrados nas Figuras 30 e 31.

De acordo com os resultados apresentados, os modelos representantes dos algoritmos SVR, LMS e M5P foram os regressores individuais mais acurados. Os três modelos selecionados são distintos quanto à natureza da aprendizagem e, portanto, o CB foi formado pelos modelos: (i) SVR_0, (ii) LMS_2 e (iii) M5P_5. Os valores dos hiper parâmetros estão apresentados no Apêndice B. Os modelos são identificados com um número para que possam ser diferenciados, em que o 0 (Zero) indica que foram usados os hiper parâmetros padrões da biblioteca Weka 3.6.10.

Figura 30 – Diagrama de caixa das amostras de erros dos modelos de regressão nas bases de validação do ISBSG



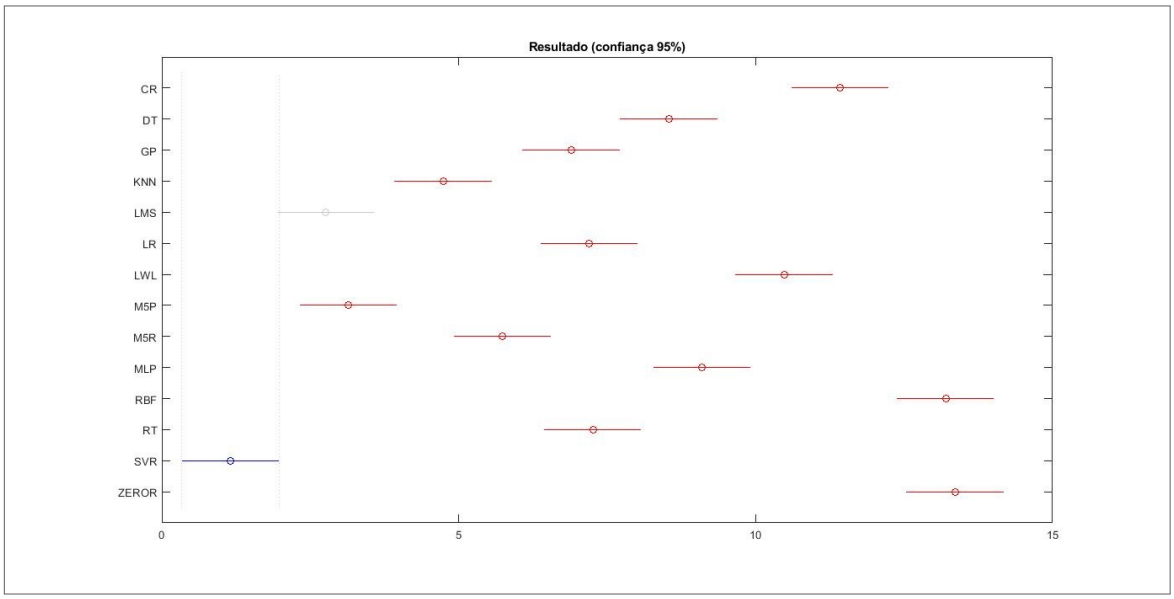
Descriptives

	SVR_0	LMS_2	M5P_5	KNN_8	M5R_2	GP_2	RT_4	LR_1	DT_3	MLP_5	LWL_0	CR_6	RBF_0	ZEROR
N	50	50	50	50	50	50	50	50	50	50	50	50	50	50
Mean	2425	2628	2690	2836	2915	2986	3094	3035	3137	3296	3441	3600	4029	4056
Median	2422	2574	2670	2772	2876	2931	3025	3041	3133	3218	3437	3548	4026	4077
Standard deviation	291	342	310	338	392	334	501	262	380	436	339	403	313	319
Range	1132	1569	1538	1639	2381	1338	2039	1189	1749	2058	1329	1546	1331	1423
Shapiro-Wilk W	0.973	0.973	0.980	0.966	0.836	0.971	0.944	0.991	0.985	0.950	0.975	0.974	0.977	0.979
Shapiro-Wilk p	0.311	0.294	0.561	0.154	< .001	0.262	0.020	0.968	0.790	0.036	0.370	0.332	0.432	0.495

Fonte: O autor (2022)

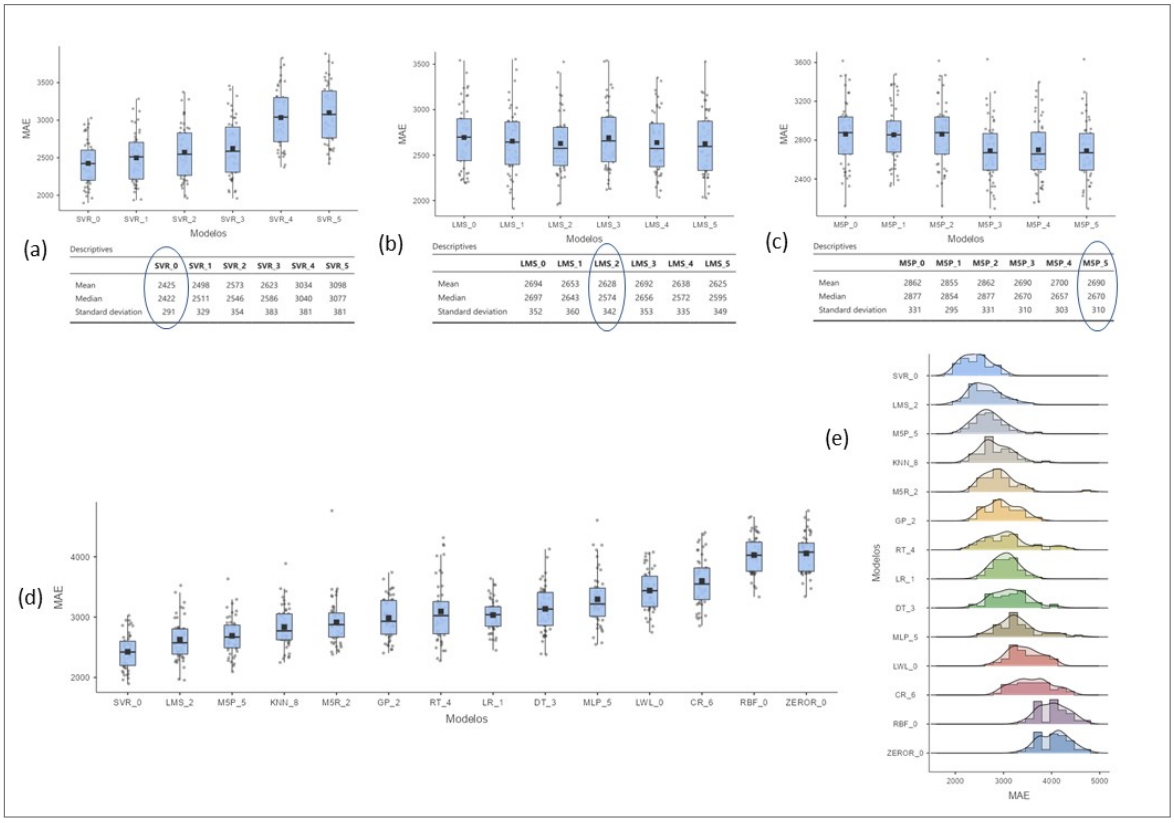
O SVR e o LMS foram algoritmos com bom desempenho, provavelmente devido à sua capacidade de lidar com ruídos (ROUSSEEUW; LEROY, 1987; FACELI et al., 2011), o que é bastante comum nos conjuntos de dados do ISBSG, conforme mostrado na Seção 5.1. Além disso, o SVR é caracterizado por uma boa capacidade de generalização e robustez para conjunto com grandes dimensionalidades (FACELI et al., 2011). Tudo isso pode ter levado os modelos criados por esses algoritmos a proporcionarem maior precisão nos resultados. O CB escolhido é composto por algoritmos com diferentes características e com melhores desempenhos para os dados do ISBSG.Mais informações sobre os algoritmos podem ser

Figura 31 – Resultado do Teste de *Friedman* e *poshoc* LSD aplicados aos erros dos modelos de regressão nas bases de validação do ISBSG



Fonte: O autor (2022)

Figura 32 – Distribuição dos erros médios absolutos, em diagramas em caixa, das variações dos modelos de regressão selecionados para: (a) modelos SVR; (b) modelos LMS; (c) modelos M5P; (d) melhores modelos de cada algoritmo e (e) melhores modelos em histograma com densidade

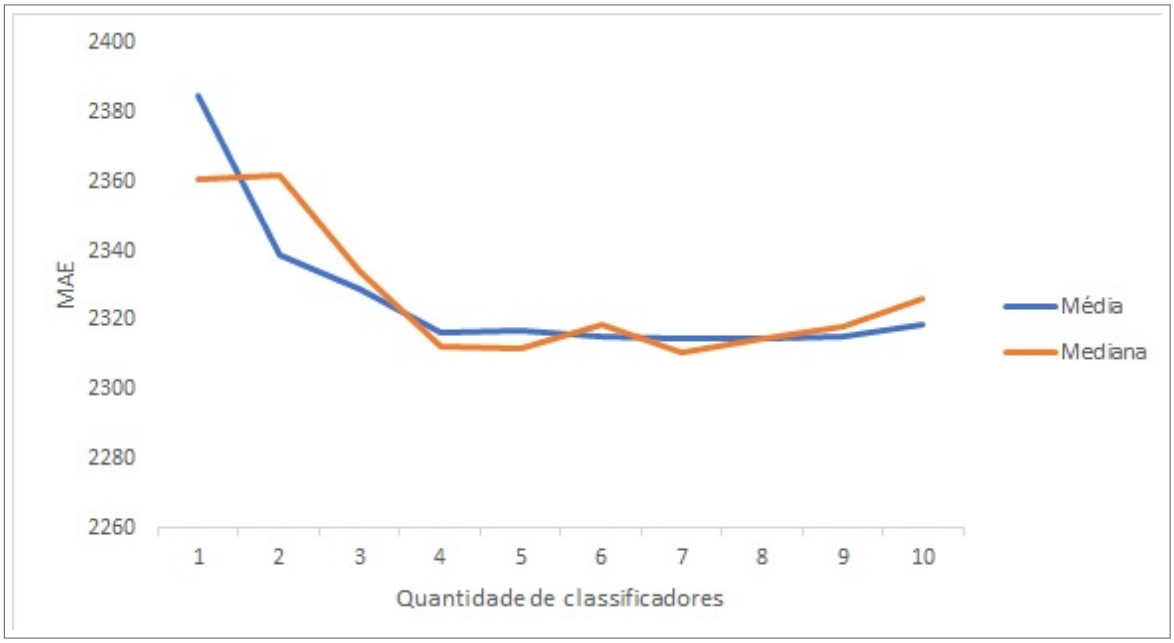


Fonte: O autor (2022)

encontradas em: SVR(SHEVADE et al., 2000); LMS (ROUSSEEUW; LEROY, 1988) e M5P (QUINLAN, 1992).

Ainda na fase de validação, também foram investigados os melhores classificadores, ou seja, os seletores dos modelos de regressão heterogêneos, entre os quais, os dez melhores na validação foram avaliados. A ordem de classificação do primeiro ao décimo classificador é baseada na menor MAE. Na Figura 33, o eixo das abcissas apresenta a quantidade de classificadores usados e, nas coordenadas, o erro absoluto médio, além da apresentação do resultado do uso de 1 a 10 classificadores. Várias combinações de classificadores seriam possíveis para o formato apresentado, no entanto, eles foram avaliados em conjunto, e na sequência do primeiro ao décimo classificador de melhor desempenho. Por exemplo, o erro atribuído ao uso de 2 classificadores é referente ao erro de utilizar o melhor e o segundo melhor classificador, qualquer outra combinação dois a dois não foi avaliada. Essa mesma lógica foi utilizada para as demais quantidades de classificadores avaliados. De acordo com o resultado apresentado na Figura 33, é perceptível que, utilizar mais que 5 classificadores, não incide, necessariamente, na melhoraria dos resultados. Desta forma, na fase de testes, foi avaliado o uso de 1 a 5 classificadores, a fim de investigar se o resultado encontrado na validação permaneceria.

Figura 33 – Média do erro absoluto médio para o uso de 1 à 10 classificadores na validação



Fonte: O autor (2022)

Os modelos KNN_1, J48_1, KSTAR, MLP_3 e RANDOM_FOREST foram, na respectiva ordem, os melhores seletores oriundos de diferentes algoritmos e, consequentemente, os modelos que compuseram o CS. A Tabela 9 apresenta os erros dos modelos de regressão selecionados dinamicamente por cada classificador do CS. De acordo com este resultado, já podemos perceber que o uso de seleção dinâmica tende a melhorar os

resultados das estimativas dos dados do ISBSG na fase de testes, uma vez que a MAE dos regressores selecionados dinamicamente pelo melhor modelo de classificação (KNN_1) (Tabela 9) foi menor do que a do melhor regressor individual (SVR_0) (Figura 30).

Outrossim, é preciso frisar que os erros da Tabela 9 não são referentes a estimativas dos classificadores, mas sim, aos modelos de regressão que foram selecionados dinamicamente por cada um deles. Apesar desses classificadores terem sido escolhidos, não é possível garantir que alcançarão bons resultados, quanto à seleção dinâmica de modelos de regressão heterogêneos, em diferentes domínios.

Tabela 9 – Média dos erros absoluto médios dos modelos de regressão selecionados dinamicamente pelos classificadores, na fase de validação, para os dados do ISBSG

Métodos	**Média**	**Des. Pad.**
KNN_1	2385	295
J48_1	2396	306
KSTAR	2403	314
MLP_3	2406	299
RANDOM FOREST	2424	285

Fonte: O autor (2022)

6.1.2 Fase de testes

Nesta seção, serão apresentadas as comparações entre os métodos oriundos do *framework* proposto e os métodos concorrentes separados por grupos, sendo os: (i) individuais (SVR, LMS e M5P); (ii) de *ensembles* estáticos e heterogêneos (Média, Mediana, Média dos Extremos, Máximo, Mínimo e *Stacking*); (iii) de *ensembles* homogêneos (*Bagging* SVR, *Bagging* LMS, *Bagging* M5P, *Boosting* SVR, *Boosting* LMS, *Boosting* M5P); (iv) de seleção dinâmica simples, considerando diferentes valores de k (*Author of the Adaptive Selection of Classifiers in Bug Predicting* (AASC-NUCCI), *Dynamic Classifier Selection By Local Accuracy* (DCS-LA) e *Dynamic Classifier Selection By Local Accuracy Weighted* (DCS-LAW)); e (v) de seleção dinâmica de múltiplos modelos com variação de k (*K-Nearest Oracle Eliminate* (KNORA-E) e *K-Nearest Oracle Union* (KNORA-U)). As versões do nosso método (PEETACO) foram avaliadas utilizando o conjunto de regressores e classificadores selecionados na fase de validação, de maneira que, o conjunto de regressores refere-se ao CB (SVR_0, LMS_2, M5P_5) e o de classificadores ao CS (KNN_1, J48_1, KSTAR, MLP_3 e RANDOM FOREST).

Após uma análise das amostras quanto à normalidade, e a confirmação através do teste de *Shapiro-wilk* de que todas elas seguem uma distribuição normal, são apresentadas, na Figura 34, as médias e os desvios padrão acumulados; foram considerados os melhores métodos avaliados em cada grupo. É possível perceber a partir do gráfico que, a média

de erros tem pequenas variações após a iteração 30. O resultado nos leva a entender que a quantidade de iterações foi suficiente para alcançarmos médias de erros estáveis. Entretanto, antes da apresentação dos testes de hipóteses, é apresentado na Figura 35 o intervalo de confiança entre as médias de erros dos melhores métodos de cada grupo.

Figura 34 – Média e desvio padrão acumulado por iteração

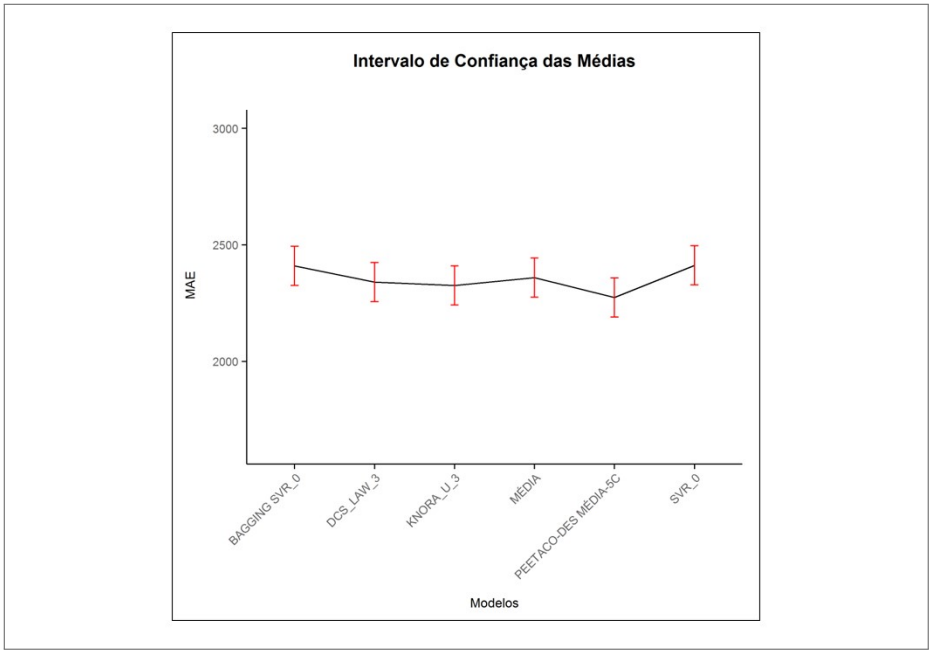
Fonte: O autor (2022)

A fim de prover uma comparação mais robusta, o teste *T-pareado* foi aplicado às médias de erros dos grupos de amostras referente a cada método. Para simplificar a análise, foi considerado, entre os métodos gerados a partir do *framework* proposto, apenas aquele que obteve o melhor resultado na fase de validação (PEETACO-DES-5C), além dos modelos concorrentes.

O teste *T-pareado* deve ser aplicado para duas amostras dependentes com distribuição normal. Uma vez que (i) a comparação foi realizada entre dois grupos de amostras, (ii) que todas as amostras de erros seguem uma distribuição normal e, (iii) que elas foram oriundas das mesmas unidades experimentais, podemos afirmar que o teste *T-pareado* é adequado ao contexto abordado.

A Tabela 10 apresenta o resultado das comparações da MAE do PEETACO-DES-5C contra os demais métodos avaliados. Na primeira coluna, temos o PEETACO-DES-5C, na segunda, os métodos concorrentes (baselines), na terceira, o resultado estatístico do teste T de *Student* e, em seguida, os graus de liberdade e o *valor-p* do teste. Nas últimas colunas são apresentados o tamanho do efeito das diferenças segundo o método de *Cohen'D* (o valor do D de *Cohen* para o teste T de *student* pareado é considerado alto se $D - Cohhen > 0,80$); e, finalmente, os resultados dos testes de normalidade de

Figura 35 – Intervalo de confiança entre as médias de erros dos melhores modelos de cada grupo de algoritmo



Fonte: O autor (2022)

Shapiro-Wilk para cada grupo de amostra. As amostras de erros do PEETACO-DES-5C obtiveram os valores $W = 0,98$ e $p - value = 0,52$, indicando assim que os dados seguem uma distribuição normal. Diante dos resultados apresentados na Tabela 10, podemos dizer que o modelo PEETACO-DES-5C obteve a MAE significativamente inferior aos demais métodos concorrentes avaliados no trabalho e que a hipótese nula foi rejeitada em todas as comparações, uma vez que o $p - value < 0,001$ indicou tal significância.

Além do teste *T-pareado*, outro teste conhecido que também pode ser aplicado ao contexto das amostras é a Análise de Variância (ANOVA). No entanto, para uma comparação par a par, é necessário um teste *post hoc*. Considerando que a quantidade de grupos de amostras é relativamente alta, é mostrado na Figura 36 o resultado da ANOVA, aplicado aos melhores métodos de cada grupo. Foi utilizado o teste (Análise de Variância para Medidas Repetidas (ANOVA-MR)) que é adequado para amostras pareadas, de maneira que é possível perceber que existem diferenças significativas entre os grupos de amostras avaliados, e que o PEETACO foi superior a todos os demais métodos. De acordo com o teste, as comparações: (i) 1º DES e 1º SD; (ii) 1º SD e 1º HETEROGÊNEO; e (iii) 1º HOMOGÊNEO e 1º INDIVIDUAL não tiveram diferenças significativas entre as médias de erros, e, consequentemente, a hipótese nula foi aceita para estes casos.

A Tabela 11 apresenta um resumo dos erros de cada método a partir de diferentes métricas e, para melhor visualização dos resultados, os melhores valores foram destacados em negrito. A descrição de cada coluna é dada a seguir, considere que os valores medidos são referentes as 50 amostras de erros.

Tabela 10 – Resultado do Teste T-pareado, aplicado às amostras dos erros absolutos médios dos métodos PEETACO-DES-5C contra os demais métodos de *baseline*, avaliados dentro dos conjuntos de testes do ISBSG

Métodos		**Teste T de Student Pareado**				**Teste de Normalidade**	
PEETACO	**Baselines**	**T**	**GL**	**p-valor**	**Tamanho do Efeito**	**W**	**p-valor**
PEETACO DES 5C							
	SVR	-9,86	49,0	< 0,001	-1,394	0,981	0,577
	LMS	-12,90	49,0	< 0,001	-1,825	0,975	0,363
	M5P	-21,45	49,0	< 0,001	-3,033	0,980	0,552
	MÉDIA	-8,00	49,0	< 0,001	-1,131	0,971	0,262
	MEDIANA	-7,59	49,0	< 0,001	-1,074	0,963	0,121
	MÉDIA DOS EXT,	-11,20	49,0	< 0,001	-1,584	0,973	0,295
	MÁXIMO	-18,73	49,0	< 0,001	-2,649	0,982	0,630
	MÍNIMO	-14,18	49,0	< 0,001	-2,005	0,975	0,360
	STACKING	-8,39	49,0	< 0,001	-1,186	0,970	0,237
	BAGGING SVR	-9,52	49,0	< 0,001	-1,347	0,974	0,333
	BAGGING LMS	-14,79	49,0	< 0,001	-2,092	0,981	0,610
	BAGGING M5P	-14,13	49,0	< 0,001	-1,998	0,976	0,388
	BOOSTING SVR	-9,85	49,0	< 0,001	-1,394	0,976	0,394
	BOOSTING LMS	-13,53	49,0	< 0,001	-1,913	0,983	0,703
	BAGGING M5P	-14,13	49,0	< 0,001	-1,998	0,976	0,388
	NUCCI	-11,47	49,0	< 0,001	-1,622	0,982	0,624
	DCS LA (k=3)	-7,04	49,0	< 0,001	-0,996	0,971	0,257
	DCS LA (k=7)	-11,37	49,0	< 0,001	-1,608	0,961	0,095
	DCS LAW (k=3)	-5,94	49,0	< 0,001	-0,839	0,985	0,773
	DCS LAW (k=7)	-8,95	49,0	< 0,001	-1,265	0,956	0,063
	KNORA E (k=3)	-7,76	49,0	< 0,001	-1,098	0,982	0,658
	KNORA E (k=7)	-8,59	49,0	< 0,001	-1,215	0,981	0,617
	KNORA U (k=3)	-5,88	49,0	< 0,001	-0,832	0,966	0,154
	KNORA U (k=7)	-7,21	49,0	< 0,001	-1,020	0,972	0,269

Fonte: O autor (2022)

Figura 36 – Resultado do teste da ANOVA-MR e *post hoc* aplicados às amostras de erros dos melhores métodos para cada grupo de algoritmos

Teste de Medidas Repetidas ANOVA - Melhores Métodos ISBSG

Within Subjects Effects

Métodos	651864	5	130373	38.3	< .001
Residual	833905	245	3404		

Note. Type 3 Sums of Squares

[3]

Between Subjects Effects

	Sum of Squares	df	Mean Square	F	p
Residual	2.61e+7	49	533615		

Note. Type 3 Sums of Squares

Post Hoc Tests

Post Hoc Comparisons - Métodos

Comparison							
Métodos		Métodos	Mean Difference	SE	df	t	p
PEETACO	-	1º DES	−72.05	9.99	49.0	−7.214	< .001
	-	1º SD	−66.11	11.14	49.0	−5.935	< .001
	-	1º HOMOGÊNEO	−135.05	14.18	49.0	−9.524	< .001
	-	1º HETEROGÊNEO	−84.99	10.63	49.0	−7.995	< .001
	-	1º INDIVIDUAL	−138.05	14.00	49.0	−9.859	< .001
1º DES	-	1º SD	5.94	14.48	49.0	0.410	0.683
	-	1º HOMOGÊNEO	−63.00	9.37	49.0	−6.726	< .001
	-	1º HETEROGÊNEO	−12.94	2.81	49.0	−4.614	< .001
	-	1º INDIVIDUAL	−66.00	9.30	49.0	−7.099	< .001
1º SD	-	1º HOMOGÊNEO	−68.94	16.13	49.0	−4.273	< .001
	-	1º HETEROGÊNEO	−18.88	14.88	49.0	−1.269	0.211
	-	1º INDIVIDUAL	−71.95	15.96	49.0	−4.509	< .001
1º HOMOGÊNEO	-	1º HETEROGÊNEO	50.05	9.35	49.0	5.355	< .001
	-	1º INDIVIDUAL	−3.01	2.66	49.0	−1.130	0.264
1º HETEROGÊNEO	-	1º INDIVIDUAL	−53.06	9.26	49.0	−5.729	< .001

Fonte: O autor (2022)

- (1) Méd.|DP: média e desvio padrão das médias dos erros absolutos de cada amostra.
- (2) Med.: mediana das médias dos erros absolutos de cada amostra.
- (3) Log.: média da média dos logaritmos dos erros absolutos.
- (4) MRE': média da média da magnitude dos erros relativos ajustados.
- (5) GR(%): média do ganho relativo em relação ao método ZERO-R.
- (6) Ran.: média do ranking do método para cada iteração.

O desempenho, em diferentes métricas, de cinco versões oriundas do *framework* proposto é apresentado na Tabela 11. A primeira versão utiliza apenas um classificador para selecionar os modelos de regressão (KNN_1). Em seguida, temos as versões com mais de um classificador, de dois até cinco classificadores foram usados como seletores. A escolha dos classificadores seguiu a ordem de desempenho alcançada na validação, conforme o resultado apresentado na Tabela 9.

A partir dos resultados da Tabela 11, podemos notar que, dos dez melhores métodos (considerando a MAE), sete são sistemas de seleção dinâmica de múltiplos modelos (*Dynamic Ensemble Selection* (DES)) e três são métodos de *Dynamic Selection* (DS) simples. De maneira semelhante ocorreu, quando foi avaliada a mediana dos erros de cada amostra, porém, a diferença nessa segunda análise foi que a MÉDIA usada como função de integração dos modelos superou o KNORA-E. Quanto ao uso do Logaritmo do erro, apenas o STACKING ficou entre os dez melhores métodos, já que os demais são métodos de seleção dinâmica. As métricas Ganho Relativo e Ranking confirmaram os resultados da primeira coluna, os nos levam a acreditar que o uso de DES inclina-se a ser adequado a projetos de Estimativa de Esforço de *Software* (EES) e possivelmente a dados semelhantes.

A fim de verificar se existem diferenças significativas entre os erros médios dos modelos de *baseline* entre si, e com a intenção de fortalecer as respostas das questões de pesquisas levantadas no Capítulo 1, as quais serão discutidas posteriormente, são apresentados, na Tabela 12, os resultados das comparações par a par dos melhores modelos concorrentes entre si, usando o teste T-Pareado. Os valores acima da diagonal principal referem-se ao valor do Teste T-Pareado. Quando o valor do teste T é negativo, indica que a média do erro absoluto dos modelos que estão identificado nas colunas foi menor do que a média do erro absoluto do respectivo modelo na linha. A diagonal principal indica em itálico a média dos erros absolutos dos modelos que estão, respectivamente, na linha e na coluna analisada. Por fim, abaixo da diagonal principal, tem-se o *valor-p* para cada comparação. Em negrito são destacadas as comparações que tiveram diferenças significativas entre as médias, considerando $\alpha = 0.05$. O resultado apresentado confirmou o que foi demonstrado nas comparações dos testes mostrados na Figura 36.

Tabela 11 – Resultado dos erros de estimativas dos métodos aplicados ao conjunto de dados do ISBSG

Métodos	**Méd. | DP**	**Med.**	**Log.**	**MRE'**	**GR(%)**	**Ran.**
SVR_0	2412 | 301	2394	2,90	1,77	40,33	16,5
LMS_2	2609 | 362	2600	2,92	1,75	35,63	24,0
M5P_5	2651 | 285	2714	2,93	1,95	34,30	25,7
MÉDIA	2359 | 310	2322 (8º)	2,89	1,53	41,68	11,7
MEDIANA	2382 | 312	2360	2,88	1,72	41,11	13,8
MÉDIA DOS EXT.	2383 | 307	2351	2,90	1,56	41,09	14,5
MÁXIMO	2654 | 283	2688	2,99	1,94	34,19	25,0
MÍNIMO	2633 | 360	2588	2,88	1,80	34,97	25,5
STACKING	2369 | 312	2334	2,87 (8º)	1,54	41,43	12,7
BAGGING SVR_0	2409 | 301	2382	2,91	1,78	40,41	16,0
BAGGING LMS_2	2590 | 329	2583	2,91	1,66	35,99	24,6
BAGGING M5P_5	2459 | **278**	2450	2,92	1,65	39,11	27,5
BOOSTING SVR_0	2413 | 302	2400	2,91	1,79	40,31	16,3
BOOSTING LMS_2	2522 | 330	2478	2,91	1,76	37,66	22,1
BOOSTING M5P_5	2844 | 347	2840	2,91	1,98	29,48	19,2
NUCCI	2371 | 306	2401	2,87 (8º)	1,51	43,76 (6º)	12,7
DCS LA (k=3)	2343 | 289 (7º)	2322 (8º)	2,86 (4º)	1,48 (7º)	42,02 (9º)	10,9 (8º)
DCS LA (k=7)	2412 | 295	2438	2,87 (8º)	1,46 (4º)	40,29	16,4
DCS LAW (k=3)	2341 | 301 (6º)	2311 (7º)	2,86 (4º)	1,48 (7º)	42,12 (8º)	10,7 (7º)
DCS LAW (k=7)	2375 | 303	2400	2,86 (4º)	1,45 (3º)	41,24	13,4
KNORA E (k=3)	2358 | 302 (10º)	2347	2,88	1,57	41,68	11,6 (10º)
KNORA E (k=7)	2366 | 299	2347	2,88	1,56	41,49	12,6
KNORA U (k=3)	2326 | 296 (5º)	2297 (4º)	2,88	1,50 (9º)	42,47 (7º)	8,4 (5º)
KNORA U (k=7)	2347 | 305 (8º)	2306 (6º)	2,88	1,50 (9º)	41,98 (10º)	9,8 (6º)
PEETACO-DS	2352 | 293 (9º)	2333 (10º)	2,87 (8º)	1,51	44,21 (5º)	11,0 (9º)
PEETACO-DES-2C	2320 | 298 (4º)	2299 (5º)	2,86 (4º)	1,47 (5º)	45,09 (4º)	7,8 (4º)
PEETACO-DES-3C	2299 | 293 (3º)	2290 (3º)	**2,85** (1º)	1,47 (5º)	45,63 (3º)	6,0 (3º)
PEETACO-DES-4C	2286 | 297 (2º)	2265 (2º)	**2,85** (1º)	**1,41** (1º)	45,93 (2º)	4,6 (2º)
PEETACO-DES-5C	**2284** | 302 (1º)	**2255** (1º)	**2,85** (1º)	1,42 (2º)	**46,06** (1º)	**3,9** (1º)

Fonte: O autor (2022)

Tabela 12 – Resultado do Teste T-pareado aplicado as amostras de erros absolutos médios dos melhores métodos de *baseline* aplicados aos dados do ISBSG

	SVR_0	**MÉDIA**	**BAG. SVR_0**	**DCS_LAW_3**	**KNORA_U_3**
SVR_0	*MAE = 2412*	**T = -5,73**	T = -1,13	**T = -4,51**	**T = -7,45**
MÉDIA	**p < 0,001**	*MAE = 2359*	**T = -5,36**	T = -1,27	**T = -5,51**
BAG. SVR_0	p = 0,26	**p < 0,001**	*MAE = 2409*	**T = -4,27**	**T = -7,28**
DCS_LAW_3	**p < 0,001**	p = 0,21	***p < 0,001***	*MAE = 2341*	T = -1,03
KNORA_U_3	**p < 0,001**	**p < 0,001**	**p < 0,001**	p = 0,31	*MAE = 2326*

Fonte: O autor (2022)

6.2 PROMISE

Nesta Seção, apresentaremos os resultados das fases de validação e testes nas bases de dados do PROMISE.

6.2.1 Fase de validação

Conforme apresentado na Seção 5.2, foram avaliadas oito bases de dados de EES do repositório PROMISE. Para cada base de dados foi realizado um processo de validação e teste semelhante ao que foi realizado no repositório do ISBSG, ou seja, foram selecionados um CB de modelos de regressão e um CS de modelos de classificação para cada base de dados. O desempenho dos modelos foi avaliado em 898 instâncias distribuídas em diferentes bases de dados, conforme mostrado na Tabela 5. Devido à avaliação dos modelos ter sido realizada por base de dados, o *leave-one-out* foi a técnica de validação usada para estes experimentos, uma vez que a maior base de dados do repositório contém 346 projetos. Nesse sentido, cada iteração avaliativa considerou $n - 1$ elementos para treinamento e 1 elemento disjunto para teste, sendo a validação realizada com os dados de treinamento. Dessa forma, teremos n iterações, sendo n o tamanho da base de dados e o tamanho dos grupos de amostras de erros de cada modelo. Considerando os mesmos parâmetros de entrada para o calcular o tamanho da amostra, as bases de dados do PROMISE com menos de 30 instâncias não foram avaliadas, posto que a menor base de dados investigada neste trabalho contém 48 instâncias.

Quanto aos algoritmos e modelos gerados, os mesmos 72 modelos oriundos dos 14 algoritmos de regressão, que foram avaliados na base de dados do ISBSG, também foram considerados neste cenário experimental. A identificação de todos os modelos é apresentada na Tabela 3. Ao fim da fase de validação, foi definido o conjunto de regressores e classificadores usados na fase de testes, conforme podemos observar na Tabela 13, que apresenta o CB e, na Tabela 16, que identifica o CS de cada base de dados, em que a primeira coluna identifica o conjunto de dados e, as demais, mostram os melhores classificadores em ordem decrescente de desempenho. Em complemento, na Tabela 14 tem-se a

média, mediana e desvio padrão, respectivamente, das amostras de erros absolutos de cada modelo e, em negrito, são marcado os menores valores em relação aos três regressores.

De acordo com os resultados apresentados na Tabela 13, é fácil perceber que os algoritmos LMS, SVR e M5P são os mais constantes dentre os regressores dos CB do PROMISE. De fato, esses também foram os mesmos algoritmos selecionados no repositório do ISBSG, no qual o SVR_0, LMS_2 e M5P_5 foram os modelos estaticamente selecionados que compuseram o CB. No entanto, a fim de aumentar o conhecimento sobre os conjuntos de dados do repositório PROMISE, foi realizada uma comparação entre os modelos individuais considerando as oito bases de dados.

Antes da comparação desses resultados, as amostras de erros de cada modelo foram analisadas quanto à normalidade do conjunto de valores. O resultado obtido através do teste de *Shapiro-Wilk* mostrou que as amostras de erros não seguiam uma distribuição normal. Dessa forma, as análises apresentadas para este repositório se basearam em métodos não paramétricos e, consequentemente, foram usados os testes de *Friedman* e da menor diferença significativa (LSD). É apresentada na Figura 37 uma avaliação geral dos algoritmos de regressão para os dados do PROMISE. Neste resultado, o algoritmo M5P foi o vencedor, seguido do LMS e do *M5 Rules* (M5R). De acordo com o resultado da Tabela 13, o M5R esteve presente em apenas dois CB de regressores, no entanto, ele foi o terceiro melhor algoritmo no contexto geral. Este fato pode ser explicado pelo resultado da Tabela 15, pois nela é apresentada a média e o desvio padrão do *ranking* de posições de cada método, considerando todas as bases de dados. O algoritmo M5R obteve a média de ranking *5,88*, enquanto o SVR ficou em seguida com a média de 6,19. Estes resultados nos levam a acreditar que o algoritmo SVR foi muito bom em algumas bases de dados e, de fato, isso aconteceu, como é mostrado na Tabela 13, mas foi mediano em outras, uma vez que ele venceu (1º lugar) em 4 de 8 conjuntos, todavia, no geral, não obteve a média de *ranking* superior a do M5R. Esta afirmação pode ser comprovada pelo valor do desvio padrão dos erros de cada modelo, já que, enquanto o SVR variou *3,80* desvios padrão, o M5R variou *3,45*. Aparentemente o M5R foi mais estável, permanecendo em média entre os melhores algoritmos.

É importante afirmar que os três melhores modelos de regressão apresentados na Figura 37 não serão usados como referência para a fase de teste, pois, a fim de deixar os resultados da fase de teste mais robustos, os conjuntos CB e CS foram selecionados para cada bases de dados, de maneira que, na avaliação final, os modelos concorrentes não se referirão a um algoritmo específico, mas aos resultados dos melhores algoritmos por base de dados. Em outras palavras, as referências serão aos três melhores modelos para cada base de dados. No entanto, esta análise é útil, e enriquece o conhecimento do leitor, quanto aos melhores algoritmos usados emEES.

No que se refere aos classificadores selecionados para o CS de cada base de dados, podemos observar na Tabela 16 que aqueles baseados em *ensembles* são mais constantes

Tabela 13 – Regressores selecionados para cada base de dados por ordem do *ranking* de posições

Bases de dados	**Regressor 1 (R1)**	**Regressor 2 (R2)**	**Regressor 3 (R3)**
China	M5P_1	M5R_0	GP_3
Cocomo81	LMS_4	M5P_4	KNN_4
Cocomonasa V1	LMS_0	M5P_5	RT_5
Cocomonasa V2	LMS_5	M5P_4	RT_5
Desharnais	SVR_2	LMS_0	M5P_1
Kitchenham	SVR_0	M5P_5	LMS_0
Maxwell	SVR_1	LMS_3	M5R_4
Myiazaki	SVR_1	LMS_5	KNN_2

Fonte: O autor (2022)

Tabela 14 – Média, Mediana e Desvio Padrão dos erros absolutos dos modelos selecionados na validação por base de dados.

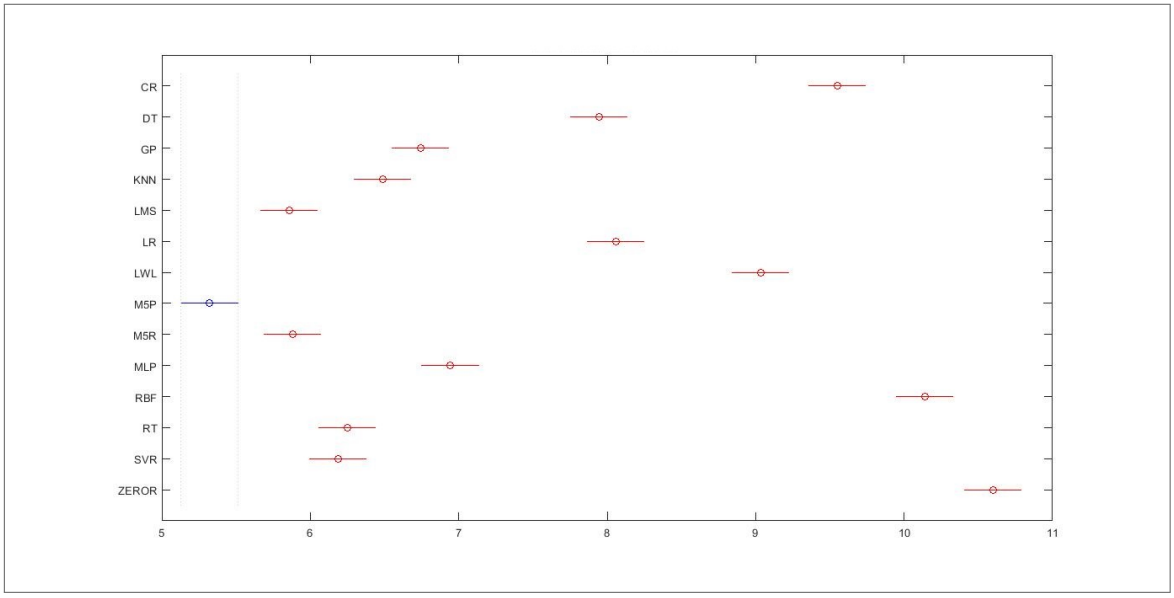
Bases de dados	**Méd. (R1| R2 | R3)**	**Med. (R1 | R2 | R3)**	**D.P. (R1 | R2 | R3)**
China	**942** | 1126 | 1360	**226** | 301 | 407	**2732** | 3501| 3934
Cocomo81	536 | 514| **506**	**62** | 100| 141	1613 | **1230** | 1359
Cocomonasa V1	**122** | 145 | 191	**28** | 54 | 42	**253** | 273 | 389
Cocomonasa V2	**347** | 359 | 440	97 | **70** | 159	848 | **725** | 745
Desharnais	1793 | **1778** | 1939	**913** | 940 | 1355	2366 | 2298 | **2027**
Kitchenham	1231 | 1432 | **1059**	**507**| 526| 560	3324 | 5300 | **1750**
Maxwell	**3070** | 3527 | 4438	**1535** | 1716 | 1652	**5767** | 5992 | 6479
Myiazaki	**39** | 47 | 53	13 | 12 | **9** |	**133** | 188 | 204

Fonte: O autor (2022)

Tabela 15 – Média e desvio padrão, respectivamente, do *Ranking* de posições dos modelos individuais na fase de validação do PROMISE

M5P	LMS	M5R	SVR	RT	KNN	GP	MLP	DT	LR	LWL	CR	RBF	ZEROR
5,32	5,86	5,88	6,19	6,25	6,49	6,74	6,94	7,94	8,06	9,03	9,55	10,14	10,60
3,85	3,75	3,45	3,80	3,40	3,89	3,62	3,30	3,67	3,77	3,89	3,94	3,20	3,76

Figura 37 – Resultado estatístico do teste de *Friedman* e *post-hoc* LSD, com 95% de confiança, aplicados aos erros amostrais dos modelos individuais na fase de validação do PROMISE



Fonte: O autor (2022)

dentre os CS das bases de dados, entretanto, os classificadores bases usados nestes *ensembles* (*Bagging* e *Boosting*) são árvores de decisão. Em seguida, também podemos observar uma forte presença dos modelos baseados em distâncias (KNN e KS) e uma árvore de decisão (RT). De maneira geral, podemos dizer que há uma diversificação de desempenho entre os classificadores usados no CS das bases de dados.

Tabela 16 – Classificadores usados em cada base de dados por ordem de acurácia

Bases de dados	**C1**	**C2**	**C3**	**C4**	**C5**
China	BA_5	AD_2	RF_0	RT_5	KS
Cocomo81	AD_2	KNN_2	KS	RF_0	RT_5
Cocomonasa V1	DT_5	BFT_2	BA_3	RT_3	KNN_6
Cocomonasa V2	DT_2	RBF_1	SVM_0	LT	KNN_7
Desharnais	AD_2	KS	RF_0	RT_5	BA_5
Kitchenham	KNN_2	AD_4	BA_4	LWL	J48_0
Maxwell	AD_2	KNN_2	RF	RT_5	KS
Myiazaki	KNN_1	RT_1	SVM_0	AD_1	DT_1

Fonte: O autor (2022)

Uma vez que os algoritmos *Adaboost*, *Bagging* e *Random Forest* estiveram presentes em diferentes bases de dados - em 6, 4 e 4 delas, respectivamente, conforme apresentado na Tabela 16 - é possível acreditar que a seleção dinâmica de modelos de regressão realizada por *ensembles* de classificadores pode ser melhor do que utilizar um classificador simples baseado em distância ou em árvores, por exemplo. No entanto, não é o que parece

acontecer, já que não é possível afirmar que os *ensembles* usados como seletores de modelos de regressão sejam os mais adequados para o contexto, visto que outros algoritmos como o REP_TREE e KNN também constituíram o CS de diferentes bases de dados. O REP_TREE apareceu em 5 das 8 bases de dados, e o KNN em 6 conjuntos. Esses foram os que estiveram mais presentes entre os classificadores que não constituem um *ensemble*, e de fato foram os mais estáveis. De maneira geral, podemos dizer que, para o contexto das bases de dados do PROMISE, algoritmos baseados em árvores de decisão e em distâncias foram os mais adequados, de acordo com o resultado da validação. De todos os classificadores selecionados e apresentados na Tabela 16 ($5 * 8 = 40$), 14 são *ensembles* de árvores de decisão, 11 são baseados em distância, 9 são árvores de decisão simples, 3 são baseados em regras e os demais estimam funções ou são probabilísticos.

Entretanto, acreditamos que utilizar apenas um classificador tende a não superar a seleção dinâmica de múltiplos modelos por diferentes critérios, pois, os experimentos realizados na fase de validação do repositório ISBSG, confirmaram essa hipótese. De acordo com os resultados do ISBSG, o uso de vários classificadores foi superior ao uso de apenas um modelo de classificação, nesse sentido, na fase de teste, realizada posteriormente, foi considerado de 1 a 5 classificadores durante a integração dos modelos. A fim de investigar esta hipótese nos dados do PROMISE, o procedimento foi repetido com até 5 classificadores.

Além disso, é possível que o tamanho, o número de atributos, as estatísticas da variável dependente, os regressores selecionados, entre outras metas-características do conjunto de dados e do próprio *framework* proposto, influenciem na definição dos melhores classificadores. Analisar a escolha desses seletores, de acordo com a base de dados, é um bom tema para estudos posteriores.

6.2.2 Fase de teste

Neste Item, serão apresentadas as comparações dos resultados da aplicação dos métodos avaliados nesta pesquisa em oito bases de dados do repositório PROMISE, sendo que esta análise difere da apresentada no Item 6.1.2, uma vez que, naquela, apenas um conjunto de dados (ISBSG) com 1466 exemplos foi avaliado. Agora, temos oito conjuntos de dados que, juntos, totalizam 898 exemplos e, nesse sentido, foram realizadas comparações por base de dados e por repositório. Diante disso, a fim de simplificar a apresentação dos resultados, as avaliações por base de dados foram resumidas em uma métrica, enquanto as comparações do teste de hipótese abordaram os 8 conjuntos de dados. As comparações das demais métricas também consideraram o repositório como um todo.

Algumas análises foram necessárias antes de definir os testes a serem realizados. A primeira é quanto à normalidade dos grupos de amostras de erros. O teste de *Shapiro-Wilk* foi usado para definir se havia normalidade nas distribuições dos dados das amostras de teste. Após a aplicação do método, todas as amostras de erros investigadas não apre-

sentaram distribuições normais. Esta conclusão foi tirada tanto a partir do resultado do teste de normalidade, quanto da análise das distribuições dos dados, as quais se comportaram de forma assimétrica. O resultado do teste apresentou o $valor - p < 0,001$ para todos os grupos de amostras, o que indica não normalidade nos dados. Nesse sentido, as comparações foram realizadas com testes não paramétricos.

A primeira avaliação foi realizada por base de dados, e, como as amostras não são normais, foi usado o *ranking* de posições por conjunto de dados. Na Tabela 17, é apresentada a média dos *rankings* dos métodos para cada conjunto de dados, a qual é obtida a partir das posições de cada método em cada ponto de teste, considerando o erro absoluto. Os menores *rankings* médios foram destacados em negrito. Quanto à paridade das amostras, podemos dizer que os grupos são pareados, uma vez que o *leave-one-out* foi a técnica de validação usada e, que os pontos de testes são os mesmos para cada método. Além disso, temos mais de 3 grupos de amostras sendo comparados entre si. Desta forma, diante das características do experimento, foi utilizado o teste de *Friedman* junto a um teste *pos hoc* para realizar as comparações estatísticas.

A fim de apresentar maior robustez nas comparações, os melhores modelos individuais para cada base de dados foram nomeados como 1º INDIVIDUAL, 2º INDIVIDUAL e 3º INDIVIDUAL, uma vez que, para cada conjunto de dados, os melhores modelos individuais se alteram entre os algoritmos, ou seja, esses métodos não se referem a um algoritmo específico, mas aos melhores modelos individuais avaliados para cada banco de dados do PROMISE. Consequentemente, essa estratégia caracteriza o resultado dos métodos individuais como uma abordagem de seleção estática simples. De maneira semelhante, são referenciados os métodos *Bagging* 1º, 2º e 3º, assim como o *Boosting* 1º, 2º e 3º. Todos os métodos concorrentes se basearam no CB de cada de base de dados para dar as respectivas saídas.

A primeira observação que podemos fazer, quanto ao resultado apresentado na Tabela 17, é quanto aos valores da média do *ranking*. Diferentemente do que aconteceu no repositório do ISBSG, a variação entre os valores médios dos *rankings* foi menor do que a apresentada na análise anterior. Por exemplo, a base de dados *Kitchenham* teve um desvio padrão entre os *rankings* dos métodos de 1,04. Esta foi a menor variação entre todas as bases de dados, sendo que a maior ocorreu no *Cocomo81* com um desvio padrão de 2,53. De fato, a diferença das médias dos rankings do melhor para o pior método no *Kitchenham* foi de 3,7. No ISBSG, essa mesma diferença foi de 23,6 (27,5 - 3,9), como pode ser verificado na na coluna *Ranking* da Tabela 11.

Esse fato ocorreu devido à técnica usada para separar os conjuntos de treino/validação e teste. O *ranking* das bases de dados do PROMISE foi baseado em um único ponto de teste (*leave-one-out*), enquanto no ISBSG o ranking foi baseado na média de erros de um grupo de exemplos de testes, visto que foi usado *hold-out* com re-amostragem. Porém, essas diferentes técnicas de validação são propensas a nos levar a resultados semelhantes,

Tabela 17 – Média dos *rankings* de cada método para cada base de dados do PROMISE, além do ranqueamento das posições gerais

Métodos	**Chi.**	**C.81**	**C.V1**	**C.V2**	**Des.**	**Kit.**	**Max.**	**Miy.**	**Pos.**
1º INDIVIDUAL	15,4	14,9	13,8	15,1	13,6	14,7	12,5	**12,4**	7º
2º INDIVIDUAL	16,9	17,1	15,5	15,2	16,0	15,3	17,1	15,3	22º
3º INDIVIDUAL	18,8	18,4	17,9	18,2	16,4	16,7	16,0	16,8	28º
MÉDIA	14,3	14,7	13,3	14,4	14,5	14,1	13,8	14,4	12º
MEDIANA	15,3	13,3	**12,5**	13,4	14,4	13,9	13,0	13,2	4º
MÉDIA DOS EXT.	14,5	15,8	15,7	15,1	15,0	14,7	14,5	15,0	17º
MÁXIMO	20,0	18,7	16,3	18,4	15,5	16,5	15,6	15,6	27º
MÍNIMO	15,8	18,4	18,4	16,7	16,0	16,3	17,0	15,6	25º
STACKING	14,6	17,7	15,2	14,1	14,9	15,3	12,1	15,0	15º
BAGGING 1º	14,5	15,7	15,1	13,2	15,2	14,8	15,9	12,8	14º
BAGGING 2º	16,5	18,2	16,1	17,3	17,3	15,3	17,8	12,8	21º
BAGGING 3º	19,1	19,3	17,4	17,3	16,7	15,4	15,8	18,7	29º
BOOSTING 1º	13,4	16,1	13,3	15,7	19,1	15,7	19,3	16,3	23º
BOOSTING 2º	16,6	15,9	14,0	15,4	18,4	16,0	16,6	16,9	24º
BOOSTING 3º	13,9	19,3	16,4	17,7	17,0	16,1	17,5	16,9	26º
NUCCI	13,8	11,6	14,1	13,8	14,3	15,7	13,7	15,6	8º
DCS LA_3	13,5	13,8	15,7	16,0	13,8	15,9	16,8	16,8	19º
DCS LA_7	13,6	13,5	17,0	16,4	12,9	16,1	16,5	16,8	20º
DCS LAW_3	13,5	13,0	15,2	16,0	14,2	16,3	16,4	17,1	18º
DCS LAW_7	13,5	13,4	15,7	15,6	13,1	15,7	16,1	17,1	16º
KNORA E_3	14,2	13,6	13,7	15,1	13,0	14,4	14,5	14,1	9º
KNORA E_7	14,1	14,0	14,5	15,2	**12,4**	14,1	14,1	14,4	10º
KNORA U_3	14,3	14,9	13,3	14,2	14,1	14,1	13,8	14,4	11º
KNORA U_7	14,3	14,7	13,3	14,4	14,5	14,1	13,8	14,4	12º
PEETACO-DS	16,0	11,8	15,3	12,9	13,9	13,7	13,7	14,8	6º
PEETACO-DES 2C	14,5	11,9	14,7	13,3	14,6	13,8	12,7	13,6	5º
PEETACO-DES 3C	13,8	11,9	14,4	13,2	14,4	13,8	**12,1**	12,5	3º
PEETACO-DES 4C	**13,0**	**11,5**	13,8	**12,6**	14,8	13,5	12,8	13,3	2º
PEETACO-DES 5C	13,2	11,7	13,8	12,8	15,1	**13,0**	13,4	12,7	1º

Fonte: O autor (2022)

quanto ao ranqueamento dos métodos. Utilizar *leave-one-out* nas bases do PROMISE é justificável porque nos permite gerar modelos mais robustos, uma vez que as bases de treinamento são maiores, pois são usados $n - 1$ exemplos para o treinamento, muito embora a variação entre os modelos criados nas iterações seja pequena.

É válido afirmar também que, o que mais interessa, para efeito de comparação, é saber a posição de cada método para cada base de dados, e isto é apresentado na Tabela 18. No resultado, é possível visualizar de forma mais clara que o PEETACO conseguiu obter bons resultados entre os conjuntos de dados. No entanto, em alguns deles, o desempenho obtido foi inferior a de alguns métodos, como é o caso, por exemplo, na base de dados *Desharnais*.

É apresentada na última coluna da Tabela 18 a média das posições de cada método e a posição de cada método no repositório. Perceba que a avaliação aplicada nas Tabelas 17 e 18 são diferentes. Apesar de terem resultados próximos, alguns métodos variaram quanto às posições, a exemplo do PEETACO-DS e da seleção dinâmica de AASC-NUCCI adaptada. Porém, a maioria dos métodos permaneceram em posições semelhantes, a exemplo dos métodos PEETACO-DES-4C/5C, que mantiveram a superioridade, quanto às médias dos rankings e posições por base de dados.

Podemos notar nas Tabelas 17 e 18 que as seis diferentes combinações do método proposto (PEETACO) venceram em 5 das 8 bases de dados, com destaque para as versões que utilizaram quatro ou cinco classificadores na seleção dinâmica. Entretanto, é importante dizer que não existe um método superior para todas as bases de dados e, portanto, aquele que tiver melhor generalização, considerando todos os conjuntos de dados, poderá ser identificado como o de melhor desempenho. Destante, de acordo com o *ranking* dos métodos por base de dados e o *ranking* geral do repositório, os métodos de múltiplos modelos baseados no *framework* proposto foram melhores do que os concorrentes avaliados, cujos resultados apresentados foram baseados na MAE.

Uma vez apresentados os resultados por base de dados, é mostrada a seguir a avaliação do repositório, considerando as 8 bases de dados. Nas comparações que seguem é apresentado o resultado da aplicação do teste de *Friedman* junto ao *pos hoc* LSD. As comparações foram realizadas por grupos de métodos, e são eles: (i) modelos individuais; (ii) modelos múltiplos baseados em *ensembles* heterogêneos; (iii) modelos múltiplos baseados em *ensembles* homogêneos; (iv) métodos de seleção dinâmica simples e (v) métodos de seleção dinâmica de múltiplos modelos. As Figuras 38, 39, 40, 41 e 42 apresentam, respectivamente, todas as comparações.

A fim de fortalecer as respostas das questões de pesquisa levantadas no início do trabalho, a Figura 43 traz uma comparação resumida entre os melhores métodos avaliados para cada grupo de métodos. Em seguida, na Tabela 19, são apresentados detalhes dos resultados do Teste de *Friedman* e *pos hoc*, com destaque para as comparações do PEETACO contra os melhores métodos de cada grupo. Considere que 1º INDIVIDUAL refere-se aos

Tabela 18 – Posição de cada método para cada base de dados do PROMISE

Métodos	**Chi.**	**C.81**	**C.V1**	**C.V2**	**Des.**	**Kit.**	**Max.**	**Miy.**	**Méd.(Pos.)**
1º INDIVIDUAL	21º	17º	9º	16º	5º	12º	3º	**1º**	15,3 (18º)
2º INDIVIDUAL	26º	22º	19º	18º	22º	16º	26º	17º	18,7 (24º)
3º INDIVIDUAL	27º	26º	28º	28º	24º	29º	19º	22º	22,8 (28º)
MÉDIA	13º	14º	4º	12º	13º	7º	10º	10º	12,5 (7º)
MEDIANA	20º	8º	**1º**	7º	12º	6º	6º	6º	11,3 (5º)
MÉDIA DOS EXT.	16º	19º	21º	14º	18º	13º	15º	15º	16,4 (21º)
MÁXIMO	29º	27º	24º	29º	21º	28º	16º	19º	22,8 (29º)
MÍNIMO	22º	25º	29º	25º	23º	27º	25º	20º	21,3 (26º)
STACKING	19º	23º	17º	10º	17º	15º	2º	16º	16,3 (20º)
BAGGING 1º	17º	18º	15º	5º	20º	14º	18º	4º	14,2 (15º)
BAGGING 2º	24º	24º	23º	8º	27º	17º	28º	4º	17,9 (23º)
BAGGING 3º	28º	28º	27º	26º	25º	18º	17º	29º	22,8 (27º)
BOOSTING 1º	3º	21º	3º	21º	29º	21º	29º	21º	13,3 (12º)
BOOSTING 2º	25º	20º	10º	19º	28º	23º	23º	25º	17,0 (22º)
BOOSTING 3º	10º	28º	25º	27º	26º	24º	27º	26º	19,7 (25º)
NUCCI	9º	2º	11º	9º	10º	19º	9º	18º	10,5 (4º)
DCS_LA_3	4º	12º	20º	23º	6º	22º	24º	22º	14,7 (17º)
DCS_LA_7	7º	10º	26º	24º	2º	25º	22º	22º	15,9 (19º)
DCS_LAW_3	5º	7º	16º	22º	9º	26º	21º	27º	13,5 (13º)
DCS_LAW_7	6º	9º	22º	20º	4º	20º	20º	27º	14,4 (16º)
KNORA_E_3	12º	11º	7º	15º	3º	11	14	9º	12,7 (9º)
KNORA_E_7	11º	13º	13º	17º	**1º**	10º	13º	13º	14,0 (14º)
KNORA_U_3	15º	16º	4º	11º	8º	7º	10º	10º	12,8 (10º)
KNORA_U_7	13º	14º	4º	12º	13º	7º	10º	10º	12,5 (7º)
PEETACO_DS	23º	4º	18º	3º	7º	3º	8º	14º	13,0 (11º)
PEETACO-DES 2C	18º	6º	14º	6º	15º	4º	4º	8º	12,3 (6º)
PEETACO-DES 3C	8º	5º	12º	4º	11º	5º	**1º**	2º	10,3 (3º)
PEETACO-DES 4C	**1º**	**1º**	8º	**1º**	16º	2º	5º	7º	7,7 (2º)
PEETACO-DES 5C	2º	3º	2º	2º	19º	**1º**	7º	3º	7,5 (1º)

Fonte: O autor (2022)

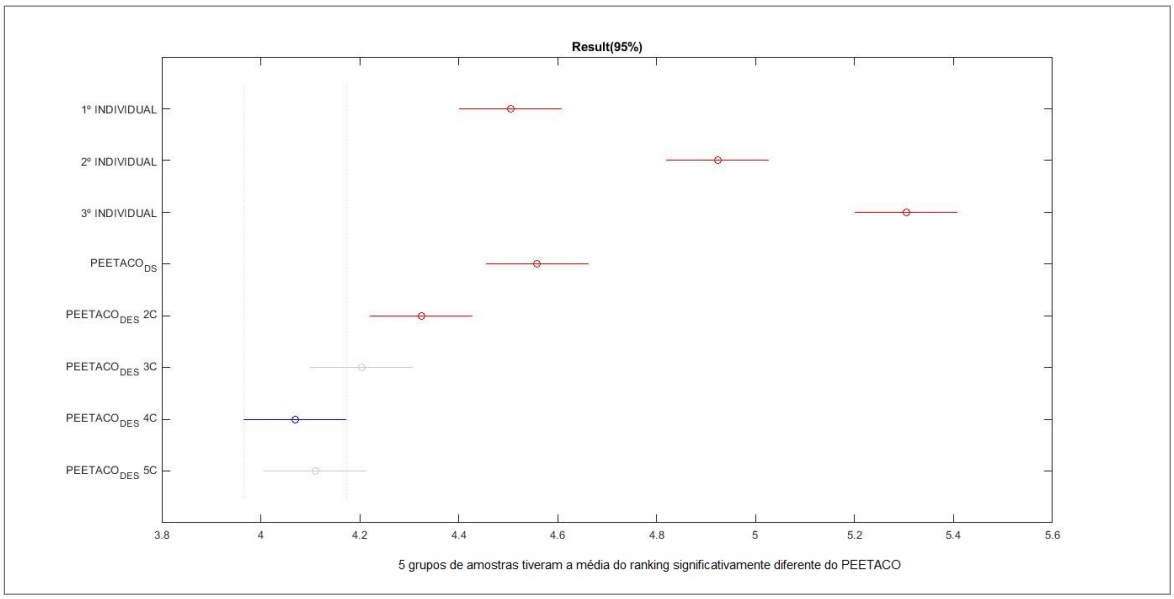
resultados do melhor modelo individual entre os participantes do CB de cada base de dados e, de maneira análoga, o 1º HETEROGÊNEO e 1º HOMOGÊNEO. O 1º DS considera o resultado do melhor método de seleção dinâmica em cada base de dados, enquanto o 1º DES utiliza os resultados do melhor método de seleção dinâmica de múltiplos modelos. Por fim, o PEETACO representa o método baseado no *framework* proposto utilizando 5 classificadores.

De acordo com os resultados, o PEETACO superou os melhores métodos de cada grupo de métodos. Entre os métodos concorrentes analisados, é possível perceber que nenhum deles foi significativamente superior a qualquer outro, no entanto, para as amostras avaliadas, a seleção estática de um sistema de múltiplos modelos heterogêneos obteve um erro médio menor do que os demais grupos de amostras. É possível que em uma amostra de dados maior, esse grupo de métodos, juntamente com os métodos de seleção dinâmica de múltiplos modelos, se sobressaíam em relação aos demais métodos concorrentes.

Após as comparações com testes de hipóteses, é apresentado o *ranking* dos métodos a partir de diferentes métricas. A Tabela 20 apresenta a média e mediana dos *rankings*, considerando inicialmente a MAE e, posteriormente, a mediana dos *rankings* das demais métricas, além das respectivas posições de cada método por métrica. Importante destacar que foi usada a mediana, devido ao fato de a distribuição dos erros não tenderem a uma distribuição normal. Nesta análise, foram considerados os 8 conjuntos de dados. O *ranking* de posições considera a posição de cada método para cada iteração na base de dados. As colunas da Tabela 20 são descritas a seguir.

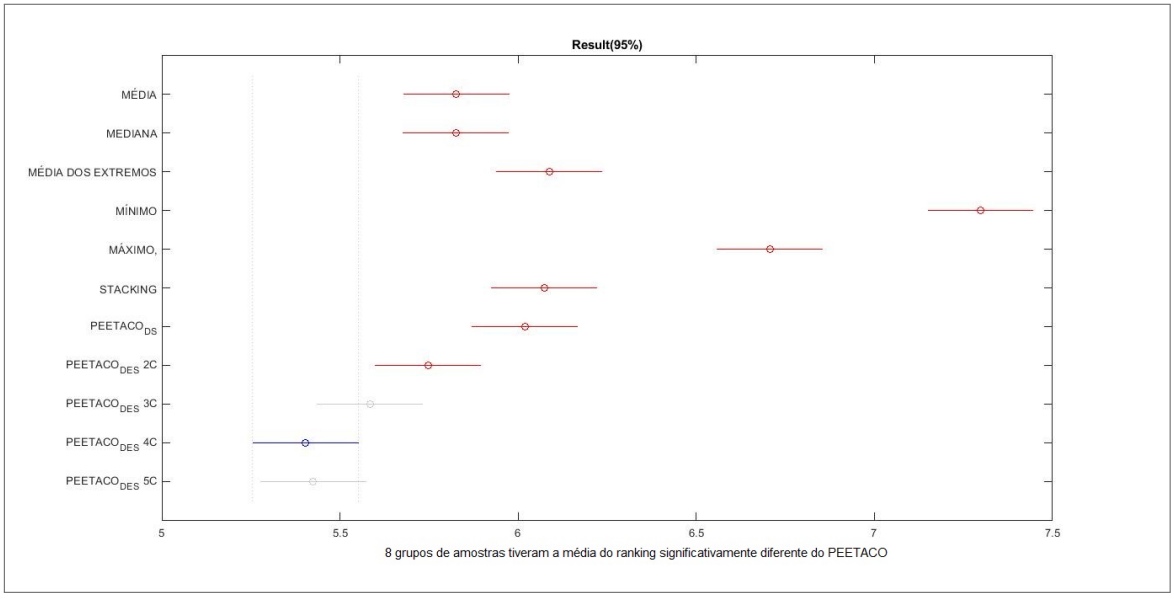
- (1) MAE: média do ranking de posições considerando o erro absoluto. É definida a posição de cada método para cada iteração e, ao fim, a média de posições é calculada.
- (2) MedAE: mediana do ranking de posições considerando o erro absoluto. Semelhante ao MAE, sendo que agora é calculado a mediana das posições.
- (3) MedLAE.: mediana do ranking de posições considerando o logaritmo do erro absoluto. É calculado o logaritmo na base10 do erro absoluto e definida a posição de cada método para cada iteração. Ao fim, é calculado a mediana dos valores.
- (4) MedMRE-adj: mediana do ranking de posições considerando magnitude do erro relativo ajustado. O MRE-adj é calculado baseado na fórmula apresentada na Equação 4.4.
- (5) MedGR(%): mediana do ranking de posições considerando o ganho relativo em relação ao método ZERO-R. O ganho relativo é definido de acordo com a Equação 4.5.
- (6) MédiaP.: média das posições de cada método por métrica.

Figura 38 – Resultado estatístico do teste de *Friedman* e *post-hoc* LSD aplicados aos (i) modelos do (PEETACO) e (ii) aos três melhores modelos individuais



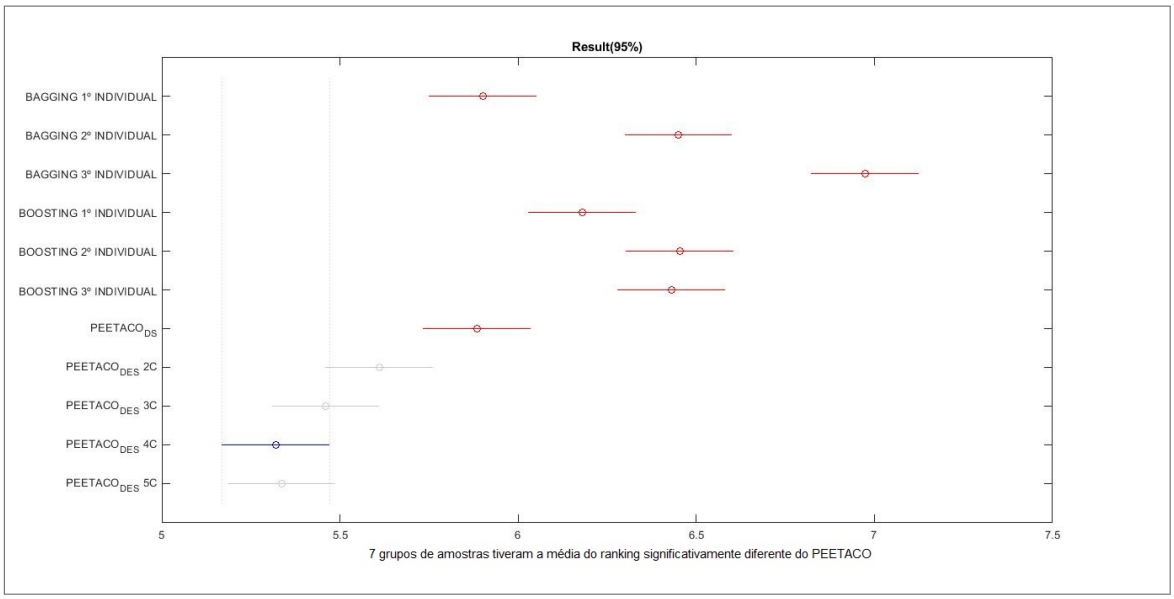
Fonte: O autor (2022)

Figura 39 – Resultado estatístico do teste de *Friedman* e *post hoc* LSD aplicados aos (i) modelos do (PEETACO) e (ii) as combinações estáticas heterogêneas



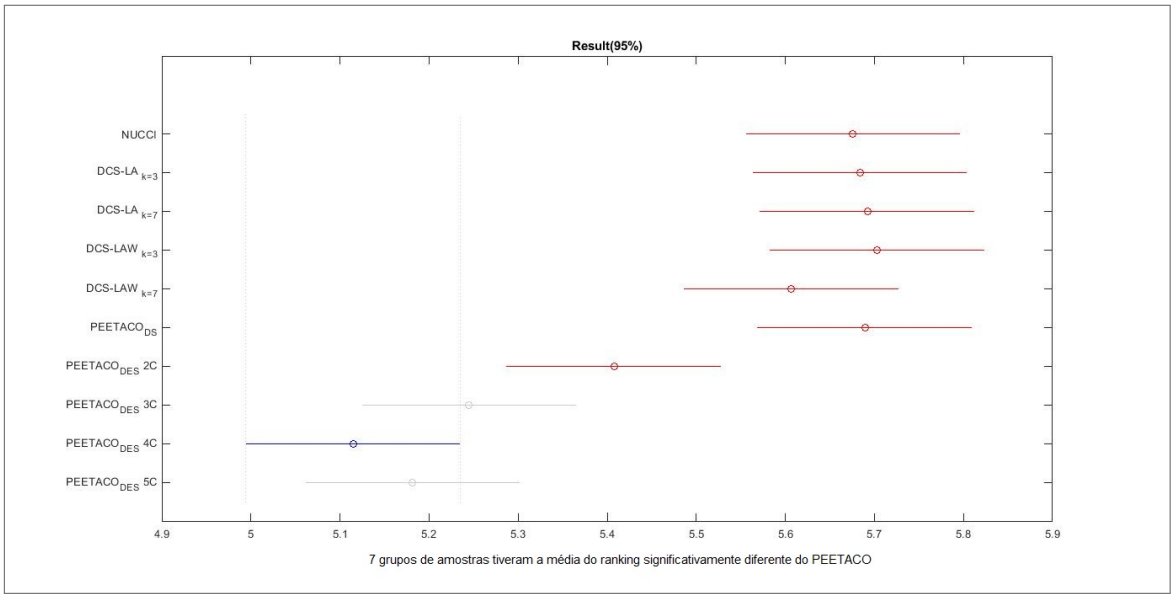
Fonte: O autor (2022)

Figura 40 – Resultado estatístico do teste de *Friedman* e *post hoc* LSD aplicados aos (i) modelos do (PEETACO) e (ii) as combinações estáticas homogêneas



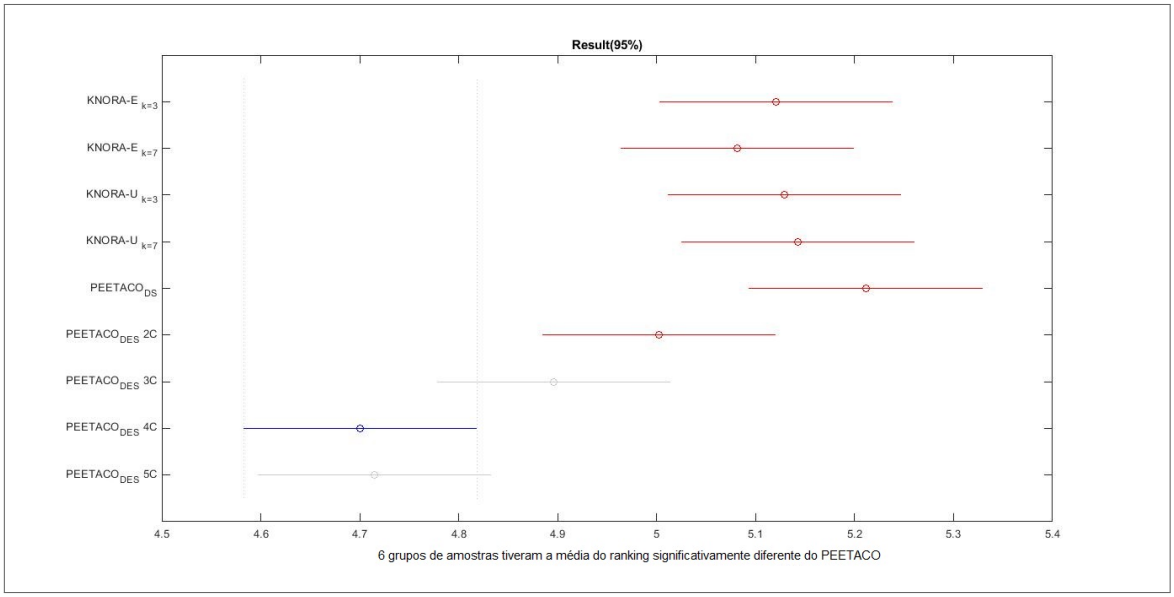
Fonte: O autor (2022)

Figura 41 – Resultado estatístico do teste de *Friedman* e *post hoc* LSD aplicados aos (i) modelos do (PEETACO) e (ii) aos métodos de seleção dinâmica simples



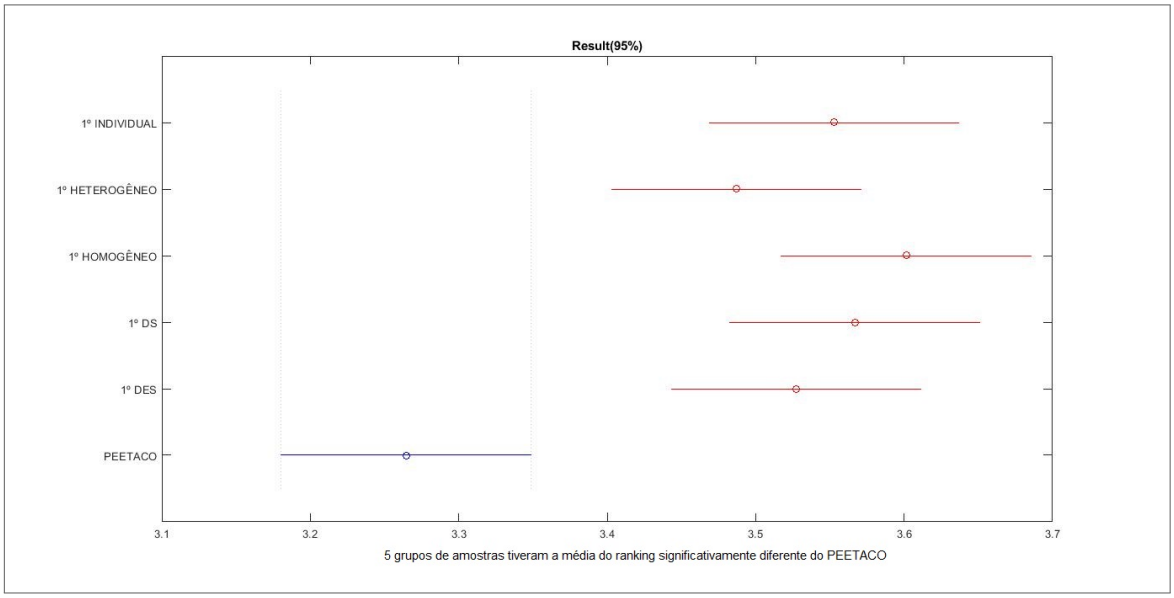
Fonte: O autor (2022)

Figura 42 – Resultado estatístico do teste de *Friedman* e *post hoc* LSD aplicados aos (i) modelos do (PEETACO) e (ii) aos métodos de seleção dinâmica de múltiplos modelos.



Fonte: O autor (2022)

Figura 43 – Resultado estatístico do teste de *Friedman* e *post hoc* LSD aplicados aos melhores métodos de cada grupo



Fonte: O autor (2022)

Tabela 19 – Detalhes do resultado estatístico do teste de *Friedman* e *post hoc* LSD aplicados aos melhores métodos de cada grupo

Friedman			
χ^2	**df**	**p**	
20.0	5	0.001	
Comparação par a par post-hoc			
		Statistic	**p**
PEETACO	**1º DES**	**3.059**	**0.002**
PEETACO	**1º DS**	**3.519**	**<.001**
PEETACO	**1º Homogêneo**	**3.920**	**<.001**
PEETACO	**1º Heterogêneo**	**2.592**	**0.010**
PEETACO	**1º Individual**	**3.357**	**<.001**
1º DES	1º DS	0.460	0.645
1º DES	1º Homogêneo	0.862	0.389
1º DES	1º Heterogêneo	0.467	0.641
1º DES	1º Individual	0.298	0.766
1º DS	1º Homogêneo	0.402	0.688
1º DS	1º Heterogêneo	0.927	0.354
1º DS	1º Individual	0.162	0.871
1º Homogêneo	1º Heterogêneo	1.328	0.184
1º Homogêneo	1º Individual	0.564	0.573
1º Heterogêneo	1º Individual	0.765	0.445

Fonte: O autor (2022)

Considerando os resultados apresentados na Tabela 18, que mostrou a posição de cada método por base de dados, e na Tabela 20, que trouxe as comparações através de diversas métricas usando o resultado de todas as bases de dados, é possível concluir que a maioria dos métodos mantiveram resultados semelhantes, enquanto alguns destoaram em relação aos resultados alcançados. Nesse sentido, podemos afirmar que a mudança de métrica pode alterar alguns resultados específicos, porém de forma pouco significativa. As principais análises observadas em relação aos resultados apresentados são:

- Os métodos gerados a partir do *framework* proposto se mantiveram entre os melhores. Considerando a análise das diferentes métricas (Tabela 20), as cinco versões avaliadas foram as melhores em um contexto geral, sendo PEETACO-DES-5C em 1º e o PEETACO-DS em 5º. Enquanto na Tabela 18, apenas o métodos PEETACO-DES-3C/4C/5C estiveram entre as melhores. O PEETACO-DES-2C perdeu 2 posições e o PEETACO-DS foi para o 11º lugar, ficando atrás das seleções estáticas de *ensembles* heterogêneos usando a média ou mediana.

- A seleção estática usando a mediana manteve um bom resultado, mesmo com a

Tabela 20 – Resultado do ranking de cada método para as bases de dados do PROMISE, a partir de diferentes métricas de avaliação

Métodos	**MAE**	**MedAE**	**MedLAE**	**MedMRE-adj**	**MedGR(%)**	**MédiaPos**
1º INDIVIDUAL	14,6 (12º)	13,5 (5º)	13,5 (5º)	14,0 (5º)	13,5 (6º)	6,6 (6º)
2º INDIVIDUAL	16,2 (24º)	17,5 (24º)	17,5 (24º)	17,5 (23º)	17,0 (23º)	23,6 (23º)
3º INDIVIDUAL	17,8 (28º)	20,5 (28º)	20,5 (28º)	20,5 (27º)	20,5 (28º)	27,8 (28º)
MÉDIA	14,2 (7º)	15,0 (13º)	15,0 (13º)	15,0 (14º)	15,0 (14º)	12,2 (14º)
MEDIANA	14,2 (7º)	14,0 (7º)	14,0 (9º)	14,0 (5º)	14,3 (9º)	7,4 (8º)
MÉDIA DOS EXT.	14,8 (18º)	15,0 (13º)	15,0 (18º)	15,0 (14º)	15,0 (14º)	15,4 (18º)
MÁXIMO	18,0 (29º)	23,8 (29º)	23,5 (29º)	22,5 (29º)	21,5 (29º)	29,0 (29º)
MÍNIMO	16,4 (26º)	20,3 (27º)	20,0 (27º)	21,0 (28º)	19,3 (27º)	27,0 (26º)
STACKING	14,8 (18º)	15,0 (13º)	17,0 (23º)	15,0 (14º)	15,0 (14º)	16,4 (20º)
1º BAGGING	14,6 (12º)	15,0 (13º)	15,0 (13º)	15,0 (14º)	16,0 (21º)	14,6 (17º)
2º BAGGING	16,1 (23º)	17,0 (22º)	16,0 (20º)	18,0 (24º)	18,0 (24º)	22,6 (22º)
3º BAGGING	17,7 (27º)	20,0 (26º)	19,0 (26º)	20,0 (26º)	19,0 (26º)	26,2 (25º)
1º BOOSTING	15,3 (21º)	15,0 (13º)	16,0 (20º)	15,0 (14º)	16,0 (21º)	17,8 (21º)
2º BOOSTING	16,3 (25º)	18,0 (25º)	18,0 (25º)	19,0 (25º)	18,0 (24º)	24,8 (24º)
3º BOOSTING	15,9 (22º)	17,0 (22º)	16,0 (20º)	14,0 (5º)	14,0 (8º)	15,4 (18º)
NUCCI	14,7 (15º)	14,0 (7º)	13,5 (5º)	14,0 (5º)	15,0 (14º)	9,2 (10º)
DCS LA (k=3)	14,7 (15º)	14,0 (7º)	14,0 (9º)	14,0 (5º)	14,3 (9º)	9,0 (9º)
DCS LA (k=7)	14,8 (18º)	14,3 (12º)	14,0 (9º)	14,5 (13º)	14,5 (11º)	12,6 (16º)
DCS LAW (k=3)	14,7 (15º)	14,0 (7º)	14,0 (9º)	14,0 (5º)	14,5 (11º)	9,4 (11º)
DCS LAW (k=7)	14,6 (12º)	14,0 (7º)	13,5 (5º)	14,0 (5º)	13,5 (6º)	7,0 (7º)
KNORA E (k=3)	14,1 (5º)	15,0 (13º)	15,0 (13º)	15,0 (14º)	15,0 (14º)	11,8 (13º)
KNORA E (k=7)	14,1 (5º)	15,0 (13º)	15,0 (13º)	15,0 (14º)	14,8 (13º)	11,6 (12º)
KNORA U (k=3)	14,2 (7º)	15,0 (13º)	15,0 (13º)	15,0 (14º)	15,0 (14º)	12,2 (14º)
KNORA U (k=7)	14,2 (7º)	15,0 (13º)	15,0 (13º)	15,0 (14º)	15,0 (14º)	12,2 (14º)
PEETACO-DS	14,5 (11º)	13,5 (5º)	13,5 (5º)	14,0 (5º)	13,0 (4º)	6,0 (5º)
PEETACO-DES-2C	13,9 (4º)	13,0 (3º)	13,0 (3º)	13,3 (4º)	13,0 (4º)	3,6 (4º)
PEETACO-DES-3C	13,5 (3º)	13,0 (3º)	13,0 (3º)	13,0 (3º)	12,3 (2º)	2,8 (3º)
PEETACO-DES-4C	**13,1 (1º)**	**12,5 (1º)**	**12,0** (1º)	**12,0** (1º)	12,5 (3º)	1,4 (2º)
PEETACO-DES-5C	13,2 (2º)	**12,5 (1º)**	**12,0 (1º)**	**12,0 (1º)**	**12,0 (1º)**	**1,2 (1º)**

Fonte: O autor (2022)

análise de diferentes métricas, mas, quando usada com a média, não foi bem avaliada, quanto às métricas adicionadas.

- Os métodos de seleção dinâmica simples foram, em geral, melhores do que as versões do KNORA para uma análise com diferentes métricas. No entanto, se formos analisar por bases de dados, o KNORA_U_7 e o KNORA_U_3 obtiveram melhores resultados.

- Os métodos que selecionam os maiores e menores valores durante as estimativas (MÁXIMO e MÍNIMO) foram, em geral, os de pior desempenho para todas as métricas. Esse fato não ocorre quando é usado o *Mean Magnitude Relative Error* (MMRE), o qual tende a favorecer as subestimativas, e prejudicar as superestimativas.

- Apesar da seleção estática do melhor modelo individual ter obtido um excelente resultado quando observado várias métricas, a seleção dos demais modelos individuais não alcançaram bons resultados. Isso nos levar a imaginar que, aumentar o número de regressores individuais, tenderá a piorar o desempenho dos sistemas de múltiplos modelos, em especial os estáticos, uma vez que todos os modelos são utilizados na fase de integração.

- Quanto aos modelos de *ensembles* homogêneos, podemos destacar o método *Boosting* que, de maneira geral, foi melhor que o *Bagging*. No entanto, ambos, quando utilizado com o melhor modelo individual, obtiveram bons resultados na metade das bases de dados avaliadas.

6.3 DADOS EDUCACIONAIS

Semelhante ao que foi apresentado nas Seções anteriores, que exibiram os resultados de validação e testes aplicados aos dados do repositório do ISBSG e do PROMISE, respectivamente, é exposto, a seguir, os resultados das fases de validação e testes dos métodos usados aplicados às bases de dados educacionais.

6.3.1 Fase de validação

Conforme apresentado na Seção 5.3, foram avaliados 6 conjuntos de dados educacionais. Para cada conjunto, foram realizadas as fases de validação e teste, e foi usada a técnica de validação *hold-out* com re-amostragem na proporção de 75% (Treinamento/Validação) e 25%(Teste). Foram selecionados um CB de modelos de regressão e um CS de modelos de classificação para cada base de dados. Os mesmos processos de avaliação realizados nas análises anteriores foram repetidos nesta. A precisão dos modelos foi avaliada por bases de dados, conforme mostrado na Tabela 5.

Quanto aos algoritmos e modelos, foram gerados 72 modelos oriundos de 14 algoritmos de regressão e, ao fim da fase de validação, foram definidos o CB e CS usados na fase de testes. A Tabela 21 apresenta o CB para cada base de dados, e as iniciais dos modelos podem ser verificadas na Tabela 3. Em complemento, na Tabela 22, tem-se a média, mediana e desvio padrão, respectivamente, das amostras de erros absolutos de cada modelo, sendo que, em negrito, são marcado os menores valores em relação aos três regressores. A Tabela 23 apresenta o CS para cada conjunto de dados, de modo que, a primeira coluna identifica o conjunto de dados, e as demais mostram os melhores classificadores em ordem decrescente de desempenho.

Tabela 21 – Regressores selecionados para cada base de dados educacional pela ordem do *ranking* de posições

Bases de dados	**Regressor (R1)**	**Regressor (R2)**	**Regressor (R3)**
Taxa Aprovação Fundamental	SVR_3	GP	M5P
Taxa Aprovação Médio	SVR_1	GP	DT_5
Taxa Evasão Fundamental	GP_3	SVR_3	M5P_4
Taxa Evasão Médio	SVR_1	KNN_8	GP_4
Taxa Reprovação Fundamental	SVR_3	GP	M5P_1
Taxa Reprovação Médio	SVR_1	LMS_2	GP

Fonte: O autor (2022)

Tabela 22 – Média, Mediana e Desvio Padrão por base de dados educacional dos erros absolutos dos modelos selecionados na validação

Bases de dados	**Méd.(R1| R2|R3)%**	**Med.(R1|R2|R3)%**	**D.P.(R1|R2|R3)%**
Tx Aprov. Fundam.	**6,98** | 7,03 | 7,10	**6,99** | 7,05 | 7,10	0,16 | **0,12** | 0,16
Tx Aprov. Médio	**4,99** | 5,11 | 5,17	**5,02** | 5,11 | 5,20	**0,26** | **0,26** | 0,30
Tx Evasão Fundam.	**2,97** | 3,00 | 3,04	**2,96** | 2,99 | 3,06	**0,12** | 0,13 | **0,12**
Tx Evasão Médio	**3,86** | 3,88 | 3,98	**3,79** | **3,79** | 3,96	0,69 | 0,67 | **0,61**
Tx Reprov. Fundam.	**4,72** | 4,75 | 4,83	**4,72** | 4,75 | 4,82	0,09 | **0,08** | 0,10
Tx Reprov. Médio	**8,70** | 8,79 | 8,93	**8,70** | 8,71 | 8,94	0,50 | 0,51| **0,41**

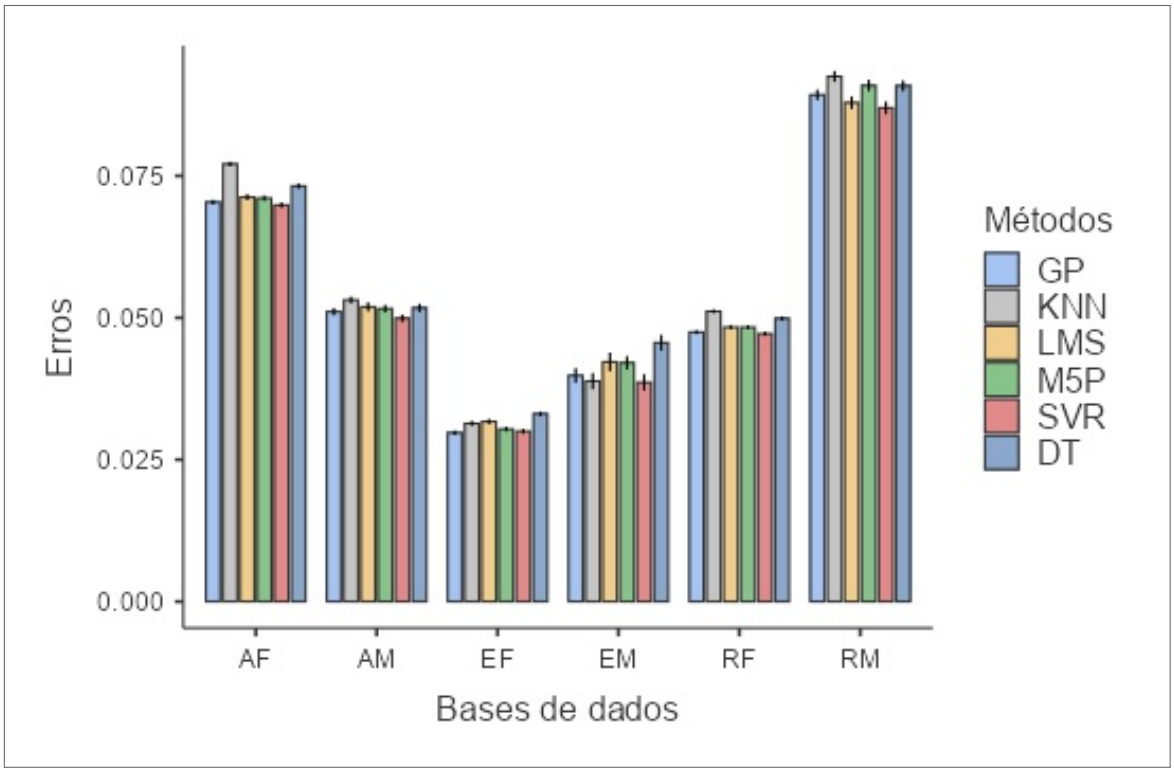
Fonte: O autor (2022)

De acordo com os resultados apresentados na Tabela 22, percebe-se que os algoritmos SVR e *Gaussian Process* (GP) são os que mais se repetem entre os regressores dos conjuntos básicos de cada bases de dados, seguidos do M5P. A fim de avaliar o desempenho dos métodos, quanto às taxas de evasão, aprovação e reprovação dos alunos, foi realizada uma comparação entre os modelos individuais na validação para cada conjunto de dados. No entanto, antes da comparação dos resultados, as amostras de erros de cada modelo para cada base de dados foram analisadas quanto à normalidade dos valores. Os

resultados obtidos através do teste de *Shapiro-Wilk* constataram que as amostras de erros tendem a seguir uma distribuição normal. Dessa forma, as análises apresentadas para este repositório se basearam em métodos paramétricos.

Na Figura 44, é apresentada uma comparação através de gráficos de barras com o intervalo de confiança sendo representado no final de cada barra. No eixo das abcissas temos as bases de dados e, nas ordenadas, a MAE. Na legenda ao lado, tem-se os seis algoritmos que constituíram o CB de ao menos um dos conjuntos de dados de educação. A comparação foi apresentada por base de dados, a fim de justificar a divisão dos dados em 6 conjuntos. De acordo com o resultado, os algoritmos SVR e GP foram os melhores em um contexto geral, seguidos do M5P e LMS e, em sequência, vieram o *K-Nearest Neighbor* (KNN) e o *Decision Trees* (DT). Neste sentido, percebe-se que os algoritmos que estimam funções apresentaram melhor desempenho para o domínio do problema.

Figura 44 – Comparação entre os principais modelos de regressão na fase de validação que utilizou os dados educacionais



Fonte: O autor (2022)

Semelhante ao que foi realizado anteriormente, cinco classificadores compuseram o CS de cada conjunto de dados. Na Tabela 23, é identificado o CS de cada conjunto de dados, em que é possível perceber que os *ensembles* novamente estão mais presentes. Ainda levando em conta esses dados, podemos destacar a presença das redes neurais e dos algoritmos baseados em distância. No entanto, de maneira geral, podemos dizer que há uma diversificação de algoritmos entre o CS de cada bases de dados educacional. No entanto, ainda não é possível estimar os algoritmos indicados para um conjunto de dados

específico, sem antes realizar um processo de validação, porém, este tipo de predição é uma abordagem que pode ser estudada em trabalhos futuros. Estimar o conjunto de seletores para uma base de dados a partir das meta-características do próprio conjunto facilitará o uso do *framework* e diminuirá o custo para alcançar o conjunto ideal. Na sessão 6.4, será apresentada uma relação inicial das bases de dados avaliadas com o CB e CS selecionados.

Tabela 23 – Classificadores usados no CS de cada base de dados de educação por ordem de acurácia

Bases de dados	**C1**	**C2**	**C3**	**C4**	**C5**
Taxa Aprovação Fundamental	LoR	NB	LMT	SVM_3	BA_3
Taxa Aprovação Médio	MLP_5	BA_3	SVM_4	AD	DS
Taxa Evasão Fundamental	AD_2	BA_2	RF_1	J48_1	KNN_1
Taxa Evasão Médio	KNN_1	RF_1	LWL	BA_5	AD_5
Taxa Reprovação Fundamental	RF_1	KNN_7	MLP_3	LMT	RBF_1
Taxa Reprovação Médio	KSTAR	KNN_4	RT_2	MLP_3	NB

Fonte: O autor (2022)

6.3.2 Fase de teste

Serão apresentadas a seguir as comparações entre as 6 bases de dados de educação, as quais diferem das anteriores porque foram utilizados vários conjuntos de dados e testes paramétricos. Apesar de existir mais de um conjunto de dados, semelhante ao PROMISE, os grupos de amostras possuem distribuições normais, conforme o resultado teste de *Shapiro-Wilk*, que apresentou $valor - p > 0,05$ para todos os grupos de amostras. Além disso, para a comparação apresentada neste Item, foi utilizada a técnica de validação *hold-out* com re-amostragem de 30 iterações a qual foi escolhida porque o tamanho das bases de dados analisadas não é adequado para o *leave-one-out* e, além disso, seguiu-se um procedimento semelhante ao aplicado no trabalho (NASCIMENTO; FAGUNDES; MACIEL, 2019), o qual utilizou os mesmos conjuntos de dados. No total, tem-se 6 conjuntos de dados que, juntos, somam 21.022 instâncias, conforme a Tabela 5. Foram realizadas comparações por conjunto de dados e uma comparação envolvendo os 6 conjuntos. Neste sentido, a fim de simplificar a apresentação dos resultados, as avaliações por base de dados foram resumidas em uma métrica, enquanto as comparações dos testes de hipóteses abordaram as 6 bases de dados como um todo, semelhante ao que foi feito com as comparações através das demais métricas.

A primeira avaliação deste domínio foi realizada por conjunto de dados. Como as amostras são normais, foi usado a média do erro absoluto e o desvio padrão de cada distribuição. Na Tabela 24, é apresentada a média e o desvio padrão do erro absoluto de cada método para cada conjunto de dados considerando as 30 iterações. A partir do resultado apresentado na Tabela 24, foi criada a Tabela 25, que identifica a posição de

cada método para cada base de dados de acordo com a média e o desvio padrão do erro absoluto. Neste formato, a análise dos melhores métodos torna-se mais simples. A fim de apresentar uma comparação mais robusta e de simplificar os resultados, foi utilizado a ANOVA para analisar o desempenho dos melhores métodos de cada grupo, ou seja, foram selecionados os resultados dos melhores métodos de cada grupo de métodos para cada base de dados. A Figura 45 apresenta os resultados destas comparações, usando os testes ANOVA e um *post hoc* separado por base de dados.

De acordo com os resultados apresentados, é possível verificar que:

- Os métodos gerados a partir do *framework* proposto alcançaram resultados aceitáveis, com destaque para os modelos com mais de três classificadores;
- De maneira geral, o PEETACO-DES-4C foi o método que obteve as melhores posições no *ranking* de base de dados, quando considerado a média e desvio padrão do erro absoluto;
- Entre os métodos de *baseline*, o *Stacking* obteve o melhor resultado e superou o PEETACO-DES com dois classificadores em 3 das 6 bases de dados;
- Na base de dados Aprovação Fundamental (AF), o PEETACO superou significativamente ($\alpha < 0,05$) todos os seus concorrentes;
- Na base de dados Aprovação Médio (AM), o PEETACO superou significativamente ($\alpha < 0,05$) os métodos de seleção dinâmica. Na comparação com a seleção estática simples, que considera o melhor modelo individual, e com o *ensemble* homogêneo do mesmo modelo, os resultados não tiveram diferenças significativas ($p - value = 0,072$);
- Na base de dados Evasão Fundamental (EF), o PEETACO superou significativamente todos os concorrentes, exceto o melhor *ensemble* heterogêneo para àquele contexto;
- Na base de dados Evasão Médio (EM), semelhante ao que aconteceu na AF, o PEETACO superou todos os métodos concorrentes;
- Na base de dados Reprovação Fundamental (RF), o PEETACO superou todos os concorrentes, mas não obteve diferença significativa em relação à seleção estática do melhor modelo heterogêneo;
- Na base de dados Reprovação Médio (RM), o PEETACO superou 3 dos 5 métodos comparados;
- O PEETACO e o *Stacking*, utilizando o CB do *framework* proposto, foram os métodos que obtiveram os melhores desempenhos nesta avaliação.

Tabela 24 – Média e desvio padrão dos erros absolutos de cada método para cada base de dados de educação

Métodos	**AF(%)**	**AM(%)**	**EF(%)**	**EM(%)**	**RF(%)**	**RM(%)**
1º INDIVIDUAL	6,90 | 0,14	5,35 | 0,30	2,95 | 0,11	3,78 | 0,55	4,66 | 0,11	8,80 | 0,45
2º INDIVIDUAL	6,93 | 0,13	5,46 | 0,27	2,99 | 0,13	3,74 | 0,57	4,66 | 0,09	8,92 | 0,51
3º INDIVIDUAL	7,03 | 0,20	5,55 | 0,28	3,04 | 0,12	3,89 | 0,54	4,76 | 0,09	9,02 | 0,39
MÉDIA	6,86 | 0,13	5,40 | 0,28	2,85 | 0,12	3,66 | 0,54	4,60 | 0,09	8,77 | 0,44
MEDIANA	6,89 | 0,12	5,42 | 0,28	**2,84 | 0,12**	3,70 | 0,54	4,60 | 0,09	8,80 | 0,45
MÉDIA DOS EXT.	6,86 | 0,13	5,40 | 0,28	2,87 | 0,12	3,66 | 0,55	4,61 | 0,09	8,78 | 0,43
MÁXIMO	6,83 | 0,14	5,33 | 0,30	3,19 | 0,12	3,98 | 0,54	4,82 | 0,08	9,06 | 0,39
MÍNIMO	7,15 | 0,12	5,62 | 0,28	2,95 | 0,13	3,71 | 0,57	4,65 | 0,11	8,88 | 0,50
STACKING	6,82 | 0,13	**5,31 | 0,29**	**2,84 | 0,12**	3,66 | 0,56	4,58 | 0,10	8,76 | 0,48
BAGGING 1º	6,90 | 0,14	5,37 | 0,31	2,95 | 0,12	3,76 | 0,54	4,66 | 0,10	8,82 | 0,44
BAGGING 2º	6,95 | 0,12	5,46 | 0,27	3,01 | 0,13	3,75 | 0,56	4,68 | 0,09	8,93 | 0,50
BAGGING 3º	6,97 | 0,12	5,53 | 0,28	2,92 | 0,12	3,89 | 0,53	4,68 | 0,09	9,01 | 0,40
BOOSTING 1º	6,87 | 0,14	5,36 | 0,30	3,23 | 0,12	3,77 | 0,54	4,61 | 0,10	8,81 | 0,44
BOOSTING 2º	6,95 | 0,13	5,57 | 0,26	2,93 | 0,14	3,75 | 0,57	4,69 | 0,09	8,92 | 0,51
BOOSTING 3º	7,02 | 0,12	5,55 | 0,28	3,23 | 0,12	4,22 | 0,57	4,75 | 0,09	9,19 | 0,41
NUCCI	6,85 | 0,13	5,37 | 0,29	2,95 | 0,12	3,73 | 0,56	4,63 | 0,09	8,91 | 0,46
DCS LA_3	6,87 | 0,13	5,40 | 0,29	2,94 | 0,12	3,74 | 0,57	4,64 | 0,09	8,83 | 0,47
DCS LA_7	6,86 | 0,13	5,37 | 0,28	2,93 | 0,11	3,74 | 0,57	4,62 | 0,09	8,79 | 0,45
DCS LAW_3	6,86 | 0,13	5,41 | 0,28	2,95 | 0,12	3,74 | 0,57	4,64 | 0,08	8,82 | 0,46
DCS LAW_7	6,85 | 0,13	5,37 | 0,29	2,93 | 0,12	3,74 | 0,57	4,63 | 0,09	8,78 | 0,45
KNORA E_3	6,86 | 0,13	5,40 | 0,28	2,89 | 0,11	3,69 | 0,57	4,63 | 0,09	8,77 | 0,45
KNORA E_7	6,86 | 0,13	5,40 | 0,28	2,90 | 0,11	3,70 | 0,56	4,63 | 0,09	8,79 | 0,45
KNORA U_3	6,86 | 0,13	5,40 | 0,28	2,85 | 0,12	3,69 | 0,54	4,60 | 0,09	8,79 | 0,46
KNORA U_7	6,86 | 0,13	5,40 | 0,28	2,85 | 0,12	3,67 | 0,54	4,60 | 0,09	8,77 | 0,44
PEETACO-DS	6,77 | 0,13	5,36 | 0,30	2,93 | 0,12	3,67 | 0,57	4,62 | 0,09	8,83 | 0,47
PEETACO-DES 2C	6,77 | 0,13	5,33 | 0,30	2,87 | 0,13	3,64 | 0,56	4,59 | 0,09	8,76 | 0,46
PEETACO-DES 3C	**6,76 | 0,13**	5,33 | 0,30	2,85 | 0,12	3,62 | 0,56	4,58 | 0,10	**8,75 | 0,46**
PEETACO-DES 4C	**6,76 | 0,13**	5,33 | 0,30	2,85 | 0,12	**3,61 | 0,57**	**4,58 | 0,09**	8,76 | 0,45
PEETACO-DES 5C	**6,76 | 0,13**	5,34 | 0,30	**2,84 | 0,12**	3,62 | 0,57	**4,58 | 0,09**	8,77 | 0,44

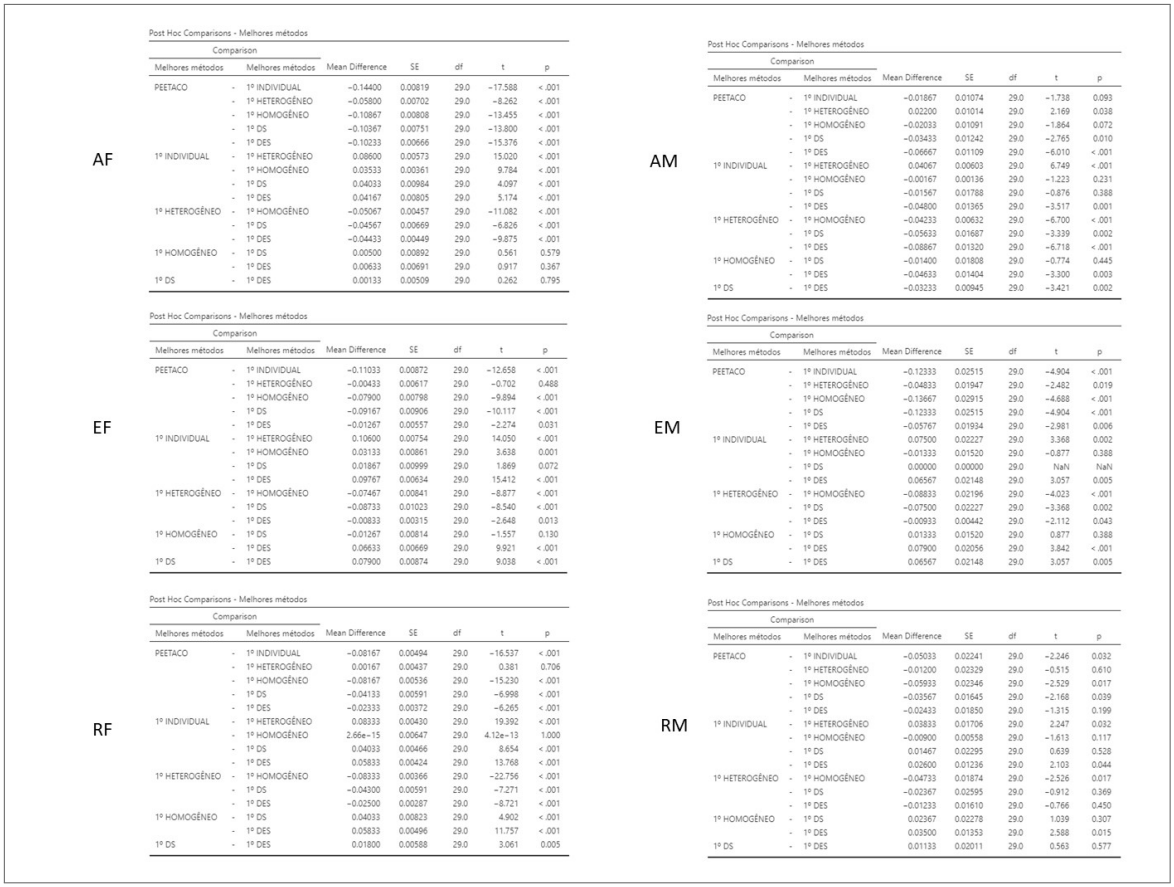
Fonte: O autor (2022)

Tabela 25 – Média do *ranking* de posições de cada método para cada base de dados de educação

Métodos	**AF**	**AM**	**EF**	**EM**	**RF**	**RM**	**Méd.**	**Med.**
1º INDIVIDUAL	21º	7º	20º	25º	22º	14º	18,17	20,5
2º INDIVIDUAL	23º	23º	24º	16º	21º	23º	21,67	23,0
3º INDIVIDUAL	28º	26º	26º	27º	28º	27º	27,00	27,0
MÉDIA	10º	14º	4º	5º	6º	5º	7,33	5,5
MEDIANA	20º	22º	**1º**	12º	10º	15º	13,33	13,5
MÉDIA DOS EXT.	10º	14º	9º	6º	6º	9º	9,00	9,0
MÁXIMO	7º	2º	27º	28º	29º	28º	20,17	27,5
MÍNIMO	29º	29º	19º	14º	20º	21º	22,00	20,5
STACKING	6º	**1º**	**1º**	7º	3º	4º	2,70	3,5
BAGGING 1º	21º	13º	20º	23º	22º	17º	19,33	20,5
BAGGING 2º	24º	23º	25º	21º	24º	25º	23,67	24,0
BAGGING 3º	26º	25º	13º	26º	25º	26º	23,50	25,5
BOOSTING 1º	19º	8º	28º	24º	10º	16º	17,50	17,5
BOOSTING 2º	24º	28º	17º	21º	26º	23º	23,17	23,5
BOOSTING 3º	27º	26º	28º	29º	27º	29º	27,67	27,5
NUCCI	8º	11º	22º	15º	14º	22º	15,33	14,5
DCS LA_3	10º	20º	18º	16º	19º	19º	17,00	18,5
DCS LA_7	10º	10º	14º	16º	12º	12º	12,33	12,0
DCS LAW_3	10º	21º	15º	16º	18º	18º	16,33	17,0
DCS LAW_7	8º	11º	15º	16º	14º	10º	12,33	12,5
KNORA E_3	10º	14º	11º	11º	14º	8º	11,33	12,5
KNORA E_7	10º	14º	12º	13º	14º	10º	12,17	12,5
KNORA U_3	10º	14º	4º	10º	6º	13º	9,50	10,0
KNORA U_7	10º	14º	4º	8º	6º	5º	7,83	7,0
PEETACO-DS	5º	8º	15º	9º	12º	19º	11,33	10,5
PEETACO-DES 2C	2º	3º	10º	4º	5º	3º	4,50	3,5
PEETACO-DES 3C	**1º**	3º	4º	2º	3º	**1º**	2,33	2,5
PEETACO-DES 4C	**1º**	3º	4º	**1º**	**1º**	2º	**2,00**	**1,5**
PEETACO-DES 5C	**1º**	6º	**1º**	3º	**1º**	5º	2,83	2,0

Fonte: O autor (2022)

Figura 45 – Comparação *post hoc* par a par entre os melhores métodos de cada base de dados de educação



AF

Post Hoc Comparisons - Melhores métodos

Comparison							
Melhores métodos		Melhores métodos	Mean Difference	SE	df	t	p
PEETACO	-	1º INDIVIDUAL	−0.14400	0.00819	29.0	−17.588	< .001
	-	1º HETEROGÊNEO	−0.05800	0.00702	29.0	−8.262	< .001
	-	1º HOMOGÊNEO	−0.10867	0.00808	29.0	−13.455	< .001
	-	1º DS	−0.10367	0.00751	29.0	−13.800	< .001
	-	1º DES	−0.10233	0.00666	29.0	−15.376	< .001
1º INDIVIDUAL	-	1º HETEROGÊNEO	0.08600	0.00573	29.0	15.020	< .001
	-	1º HOMOGÊNEO	0.03533	0.00361	29.0	9.784	< .001
	-	1º DS	0.04033	0.00984	29.0	4.097	< .001
	-	1º DES	0.04167	0.00805	29.0	5.174	< .001
1º HETEROGÊNEO	-	1º HOMOGÊNEO	−0.05067	0.00457	29.0	−11.082	< .001
	-	1º DS	−0.04567	0.00669	29.0	−6.826	< .001
	-	1º DES	−0.04433	0.00449	29.0	−9.875	< .001
1º HOMOGÊNEO	-	1º DS	0.00500	0.00892	29.0	0.561	0.579
	-	1º DES	0.00633	0.00691	29.0	0.917	0.367
1º DS	-	1º DES	0.00133	0.00509	29.0	0.262	0.795

AM

Post Hoc Comparisons - Melhores métodos

Comparison							
Melhores métodos		Melhores métodos	Mean Difference	SE	df	t	p
PEETACO	-	1º INDIVIDUAL	−0.01867	0.01074	29.0	−1.738	0.093
	-	1º HETEROGÊNEO	0.02200	0.01014	29.0	2.169	0.038
	-	1º HOMOGÊNEO	−0.02033	0.01091	29.0	−1.864	0.072
	-	1º DS	−0.03433	0.01242	29.0	−2.765	0.010
	-	1º DES	−0.06667	0.01109	29.0	−6.010	< .001
1º INDIVIDUAL	-	1º HETEROGÊNEO	0.04067	0.00603	29.0	6.749	< .001
	-	1º HOMOGÊNEO	−0.00167	0.00136	29.0	−1.223	0.231
	-	1º DS	−0.01567	0.01788	29.0	−0.876	0.388
	-	1º DES	−0.04800	0.01365	29.0	−3.517	0.001
1º HETEROGÊNEO	-	1º HOMOGÊNEO	−0.04233	0.00632	29.0	−6.700	< .001
	-	1º DS	−0.05633	0.01687	29.0	−3.339	0.002
	-	1º DES	−0.08867	0.01320	29.0	−6.718	< .001
1º HOMOGÊNEO	-	1º DS	−0.01400	0.01808	29.0	−0.774	0.445
	-	1º DES	−0.04633	0.01404	29.0	−3.300	0.003
1º DS	-	1º DES	−0.03233	0.00945	29.0	−3.421	0.002

EF

Post Hoc Comparisons - Melhores métodos

Comparison							
Melhores métodos		Melhores métodos	Mean Difference	SE	df	t	p
PEETACO	-	1º INDIVIDUAL	−0.11033	0.00872	29.0	−12.658	< .001
	-	1º HETEROGÊNEO	−0.00433	0.00617	29.0	−0.702	0.488
	-	1º HOMOGÊNEO	−0.07900	0.00798	29.0	−9.894	< .001
	-	1º DS	−0.09167	0.00906	29.0	−10.117	< .001
	-	1º DES	−0.01267	0.00557	29.0	−2.274	0.031
1º INDIVIDUAL	-	1º HETEROGÊNEO	0.10600	0.00754	29.0	14.050	< .001
	-	1º HOMOGÊNEO	0.03133	0.00861	29.0	3.638	0.001
	-	1º DS	0.01867	0.00999	29.0	1.869	0.072
	-	1º DES	0.09767	0.00634	29.0	15.412	< .001
1º HETEROGÊNEO	-	1º HOMOGÊNEO	−0.07467	0.00841	29.0	−8.877	< .001
	-	1º DS	−0.08733	0.01023	29.0	−8.540	< .001
	-	1º DES	−0.00833	0.00315	29.0	−2.648	0.013
1º HOMOGÊNEO	-	1º DS	−0.01267	0.00814	29.0	−1.557	0.130
	-	1º DES	0.06633	0.00669	29.0	9.921	< .001
1º DS	-	1º DES	0.07900	0.00874	29.0	9.038	< .001

EM

Post Hoc Comparisons - Melhores métodos

Comparison							
Melhores métodos		Melhores métodos	Mean Difference	SE	df	t	p
PEETACO	-	1º INDIVIDUAL	−0.12333	0.02515	29.0	−4.904	< .001
	-	1º HETEROGÊNEO	−0.04833	0.01947	29.0	−2.482	0.019
	-	1º HOMOGÊNEO	−0.13667	0.02915	29.0	−4.688	< .001
	-	1º DS	−0.12333	0.02515	29.0	−4.904	< .001
	-	1º DES	−0.05767	0.01934	29.0	−2.981	0.006
1º INDIVIDUAL	-	1º HETEROGÊNEO	0.07500	0.02227	29.0	3.368	0.002
	-	1º HOMOGÊNEO	−0.01333	0.01520	29.0	−0.877	0.388
	-	1º DS	0.00000	0.00000	29.0	NaN	NaN
	-	1º DES	0.06567	0.02148	29.0	3.057	0.005
1º HETEROGÊNEO	-	1º HOMOGÊNEO	−0.08833	0.02196	29.0	−4.023	< .001
	-	1º DS	−0.07500	0.02227	29.0	−3.368	0.002
	-	1º DES	−0.00933	0.00442	29.0	−2.112	0.043
1º HOMOGÊNEO	-	1º DS	0.01333	0.01520	29.0	0.877	0.388
	-	1º DES	0.07900	0.02056	29.0	3.842	< .001
1º DS	-	1º DES	0.06567	0.02148	29.0	3.057	0.005

RF

Post Hoc Comparisons - Melhores métodos

Comparison							
Melhores métodos		Melhores métodos	Mean Difference	SE	df	t	p
PEETACO	-	1º INDIVIDUAL	−0.08167	0.00494	29.0	−16.537	< .001
	-	1º HETEROGÊNEO	0.00167	0.00437	29.0	0.381	0.706
	-	1º HOMOGÊNEO	−0.08167	0.00536	29.0	−15.230	< .001
	-	1º DS	−0.04133	0.00591	29.0	−6.998	< .001
	-	1º DES	−0.02333	0.00372	29.0	−6.265	< .001
1º INDIVIDUAL	-	1º HETEROGÊNEO	0.08333	0.00430	29.0	19.392	< .001
	-	1º HOMOGÊNEO	2.66e−15	0.00647	29.0	4.12e−13	1.000
	-	1º DS	0.04033	0.00466	29.0	8.654	< .001
	-	1º DES	0.05833	0.00424	29.0	13.768	< .001
1º HETEROGÊNEO	-	1º HOMOGÊNEO	−0.08333	0.00366	29.0	−22.756	< .001
	-	1º DS	−0.04300	0.00591	29.0	−7.271	< .001
	-	1º DES	−0.02500	0.00287	29.0	−8.721	< .001
1º HOMOGÊNEO	-	1º DS	0.04033	0.00823	29.0	4.902	< .001
	-	1º DES	0.05833	0.00496	29.0	11.757	< .001
1º DS	-	1º DES	0.01800	0.00588	29.0	3.061	0.005

RM

Post Hoc Comparisons - Melhores métodos

Comparison							
Melhores métodos		Melhores métodos	Mean Difference	SE	df	t	p
PEETACO	-	1º INDIVIDUAL	−0.05033	0.02241	29.0	−2.246	0.032
	-	1º HETEROGÊNEO	−0.01200	0.02329	29.0	−0.515	0.610
	-	1º HOMOGÊNEO	−0.05933	0.02346	29.0	−2.529	0.017
	-	1º DS	−0.03567	0.01645	29.0	−2.168	0.039
	-	1º DES	−0.02433	0.01850	29.0	−1.315	0.199
1º INDIVIDUAL	-	1º HETEROGÊNEO	0.03833	0.01706	29.0	2.247	0.032
	-	1º HOMOGÊNEO	−0.00900	0.00558	29.0	−1.613	0.117
	-	1º DS	0.01467	0.02295	29.0	0.639	0.528
	-	1º DES	0.02600	0.01236	29.0	2.103	0.044
1º HETEROGÊNEO	-	1º HOMOGÊNEO	−0.04733	0.01874	29.0	−2.526	0.017
	-	1º DS	−0.02367	0.02595	29.0	−0.912	0.369
	-	1º DES	−0.01233	0.01610	29.0	−0.766	0.450
1º HOMOGÊNEO	-	1º DS	0.02367	0.02278	29.0	1.039	0.307
	-	1º DES	0.03500	0.01353	29.0	2.588	0.015
1º DS	-	1º DES	0.01133	0.02011	29.0	0.563	0.577

Fonte: O autor (2022)

6.4 DISCUSSÃO

Nesta seção, serão apresentadas discussões acerca dos resultados constatados nas Seções 6.1 e 6.2. Os testes experimentais realizados darão suporte às respostas das questões de pesquisa abordadas. Os resultados foram separados por repositório, mas a análise que segue será apresentada de maneira geral. As questões de pesquisa 01, 02 e 03 referem-se ao contexto de Aprendizagem de Máquina (AM) aplicado à EES, enquanto as questões de pesquisa 04 e 05 visam a um contexto geral.

6.4.1 Questão de pesquisa 01

Os diferentes tipos de *ensembles*, realmente, superam os métodos individuais em problemas de EES e *Education Data Mining* (EDM)?

Iniciando pela análise das Figuras 36 (ISBSG) e 38 (PROMISE), podemos ver que o PEETACO, especialmente o PEETACO-DES, foi superior aos modelos individuais. As Tabelas 10 e 11 fortaleceram essa afirmativa com o teste *T-pareado* e o uso de diferentes métricas, respectivamente. Neste sentido, podemos dizer que construir Sistema de Múltiplos Modelos (SMM) melhora os resultados obtidos pelos métodos individuais em

problemas de EES. Esta afirmação pode ser fortalecida pelo resultado da Tabela 12 que mostrou a superioridade da seleção estática e dinâmica de *ensembles* heterogêneos em relação à seleção estática simples (melhor individual).

Adicionalmente, a Tabela 18 mostra uma superioridade, quanto aos *rankings* dos *ensembles* estáticos heterogêneos (MÉDIA e MEDIANA) e das versões do KNORA em relação aos métodos individuais, considerando o repositório do PROMISE. Por fim, a Figura 43 e a Tabela 19 também fortalecem essas melhorias, no entanto, sem diferenças significativas, quando considerada a seleção estática de *ensemble* heterogêneo ou homogêneo. Assim, além dos *ensembles*, a seleção dinâmica também tem uma forte tendência a melhorar a precisão das previsões individuais em EES. No entanto, os resultados podem variar de acordo com as bases de dados e com os critérios de seleção utilizados. Todos esses resultados corroboram os levantamento realizado no Capítulo 3, em que, a maioria dos estudos que utilizaram SMM apresentaram uma melhora nos resultados alcançados pelos modelos individuais.

Para os dados educacionais, o avanço obtido com o uso de SMM também foi significativo. O melhor desempenho de um modelo individual no contexto educacional foi no conjunto Aprovação Médio, em que o melhor modelo individual alcançou o 7º lugar, conforme apresenta a Tabela 25. Este desempenho foi abaixo dos obtidos nos repositórios de EES, o que pode ser confirmado pelo resultado apresentado na Figura 45, que mostra o PEETACO sendo significativamente superior em 5 das 6 bases de dados.

6.4.2 Questão de pesquisa 02

Qual o desempenho do *framework* proposto e dos diferentes tipos de *ensembles* em problemas de EES e EDM?

Uma vez concluído que *ensembles* estáticos e dinâmicos tendem a superar os métodos individuais, a análise seguinte será referente à comparação entre os diferentes tipos de *ensembles*. Quanto aos *ensembles* homogêneos e heterogêneos avaliados dentro do contexto desta pesquisa, percebe-se uma melhoria alcançada pelos heterogêneos. Considerando inicialmente os resultados do ISBSG, na Figura 35, a melhor seleção estática usando um *ensemble* heterogêneo (MÉDIA) foi mais eficaz do que o BAGGING+SVR_0, que foi o melhor método usando *ensemble* homogêneo. Essa superioridade foi confirmada com o teste ANOVA apresentado na Figura 36. A Tabela 11 também favorece essa conclusão, uma vez que confirmou o resultado com o uso de diferentes métricas, no entanto, esta análise considerou apenas o resultado do ISBSG. Nos dados do repositório do PROMISE, a superioridade dos *ensembles* heterogêneos em relação aos homogêneos se manteve, desta vez através da MEDIANA, enquanto o BAGGING continuou sendo o melhor *ensemble* homogêneo. De maneira geral, para os dados do PROMISE e do ISBSG, o BAGGING foi melhor do que o BOOSTING, porém, quando a análise é realizada nos dados educacionais, o STACKING usando o CB do *framework* foi o melhor *ensemble* estático. O STACKING

se destacou em todos os conjuntos de dados educacionais, vencendo em 2 dos 6 conjuntos e obtendo a média do *raking* de posições semelhante as do PEETACO.

A comparação dos *ensembles* estáticos e dinâmicos será abordada através de duas perspectivas, em que, a primeira, considera o uso dos métodos conhecidos na literatura e, a segunda, através do *framework* proposto neste trabalho. Iniciando, a partir do resultado apresentado na Figura 35 (ISBSG), podemos ver que o KNORA (*ensemble* dinâmico) obteve uma melhoria em relação à MÉDIA (*ensemble* estático), e que, conforme apresentado na Figura 36, foi significativa. A Tabela 11 corrobora essa análise, apresentando o KNORA_U entre os melhores métodos para o contexto do ISBSG, mesmo a partir de diferentes métricas de avaliação. No obstante, estes resultados não se repetiram no contexto do PROMISE, pois, neste repositório, a seleção estática usando a MEDIANA, de maneira geral, foi melhor do que todas as versões do KNORA, conforme podemos ver na Tabela 18, apesar deste não ter sido um avanço significativo por parte do *ensemble* estático (MEDIANA), conforme apresentado na Figura 43 e detalhado na Tabela 19.

Diante dos resultados apresentados e, partindo do contexto desta pesquisa, o método KNORA, implementado para problemas de regressão e da forma abordada neste trabalho, pôde alcançar melhorias em relação aos *ensembles* heterogêneos mas, em geral, os resultados não foram significativos. Em especial, o KNORA_U foi a melhor versão do KNORA para problemas de EES. De forma semelhante, a SMM também se destacou no contexto dos dados educacionais através do KNORA_U. Considerando a primeira abordagem de comparação, o método superou os *ensembles* homogêneos e parte dos heterogêneos, sendo superado pelo STACKING e pela MÉDIA das saídas individuais. Além do mais, quanto às funções de integração lineares, a MEDIANA inclina-se a ser mais estável no domínio de EES e a MÉDIA para os dados educacionais. Nesse sentido, a partir dos resultados obtidos, não foi possível concluir a melhor estratégia de construção de *ensembles*, considerando a seleção dinâmica por localidade e dentro do contexto deste trabalho.

Partindo para a segunda perspectiva de análise, a dos métodos gerados a partir do *framework* proposto e considerando o repositório do ISBSG, constata-se que o PEETACO-DES 5C foi melhor do que a seleção estática de *ensembles* heterogêneos e também superior aos homogêneos, conforme mostra o resultado da Figura 35. Dessa maneira, podemos dizer que a melhoria alcançada foi significativa, baseado no resultado da Figura 36 e da Tabela 10. Analisando os resultados nas bases do PROMISE, podemos ver que estas melhorias se repetem, conforme mostra a Figura 39, que apresentou superioridade do PEETACO-DES 5C/4C em relação a todos os *ensembles* estáticos heterogêneos. As versões do PEETACO-DES 2C/3C não superaram significativamente os *ensembles* estáticos, apesar de terem alcançado melhores desempenhos no experimento realizado - de acordo com os resultados das Tabelas 18 e 20. Por fim, o PEETACO também venceu em 5 das 6 bases de dados educacionais, tendo o PEETACO-DES 4C apresentado o melhor desempenho.

A superioridade dos métodos gerados a partir do *framework* proposto também pode

ser observada, quando comparada às combinações homogêneas. A Figura 36 mostra que o PEETACO-DES superou o melhor método de *ensemble* homogêneo e, o resultado apresentado na Tabela 10 apresenta mais detalhes dessa superioridade. Esses resultados foram fortalecidos com as comparações realizadas através de diferentes métricas, de acordo com o que foi apresentado na Tabela 11. Neste sentido, é possível dizer que as combinações heterogêneas, juntamente com a seleção dinâmica de *ensembles*, a partir de diferentes critérios, justificam a melhoria alcançada. As combinações homogêneas dos modelos de regressão usados neste trabalho não apresentaram avanços significativos em relação aos demais modelos combinados no contexto da EES, entretanto, de maneira geral, os *ensembles* homogêneos foram melhores do que os seus respectivos modelos bases utilizados individualmente. De fato, essas combinações normalmente superam os seus modelos bases em EES (IDRI; HOSNI; ABRAN, 2016a). Para os dados educacionais, a superioridade dos ensembles homogêneos se repetiu apenas para o BOOSTING, que foi o melhor método homogêneo para o domínio dos dados educacionais.

Considerando os *ensembles* heterogêneos, o PEETACO-DES 5C venceu em 7 das 9 bases de dados de EES, além de ter superado os *ensembles* homogêneos em todos os testes. Para os dados educacionais, o PEETACO foi superior ao melhor *ensemble* heterogêneo em 2 das 6 bases de dados (AF e EM), porém, foi superado por uma delas (AM). Por outro lado, o STACKING foi o maior adversário do *framework* proposto no domínio educacional. Os ensembles homogêneos foram superados em todas as base de dados educacionais, sendo significativamente em 5 das 6 investigadas. Diante do que já foi apresentado, os resultados indicam que o desempenho obtido por combinações heterogêneas pode ser melhorado através da seleção dinâmica de *ensembles*, utilizando diferentes critérios. As discussões abordadas nos parágrafos anteriores dão suporte a esta afirmação, uma vez que as versões do PEETACO-DES conseguiram alcançar melhores desempenhos, quando comparadas às seleções estáticas homogêneas e heterogêneas dentro do contexto deste trabalho.

6.4.3 Questão de pesquisa 03

Qual o desempenho do *framework* proposto e dos métodos de seleção dinâmica no contexto de EES e EDM?

Iniciando a partir da seleção dinâmica simples, para os dados do ISBSG, o PEETACO-DS foi melhor do que os *ensembles* estáticos heterogêneos, mas foi inferior, de maneira geral, no PROMISE. Isso nos dar evidências iniciais de que o uso de vários critérios para selecionar modelos de regressão pode levar a melhorias, quanto à precisão das estimativas, visto que o PEETACO-DES superou os *ensembles* estáticos, conforme já foi comentado. A Figura 41 apresenta um resultado que considera todas as bases de dados do PROMISE, e, assim, é possível perceber que apenas o PEETACO-DS não superou significativamente os métodos de *ensembles* homogêneos. O uso de seleção dinâmica simples, de maneira geral, considerando outros métodos conhecidos, não foi suficiente para superar todos os

ensembles estáticos heterogêneos. A Figura 43 e a Tabela 19 reforçam os resultados comparando apenas os métodos que alcançaram os melhores resultados por grupo de métodos. Nos dados educacionais, os métodos de seleção dinâmica simples não obtiveram um bom desempenho, conforme mostra a Tabela 25. O PEETACO-DS foi o melhor entre os métodos de DS, o que fortalece a proposta de utilizar classificadores para seleção dinâmica, mesmo ele tendo sido superado pelos *ensembles* heterogêneos.

Foram avaliados cinco métodos de DS simples, considerando diferentes valores de k para o DCS-LA e DCS-LAW. Apesar da superioridade de todas as versões do PEETACO-DES, o PEETACO-DS não superou todos os métodos de DS simples, uma vez que foi superado por dois métodos avaliados, exceto para a métrica ganho relativo, conforme mostra a Tabela 11. De fato, os métodos de DS simples tiveram bons desempenho nos dados do ISBSG, superando inclusive algumas versões do *K-Nearest Oracle* (KNORA), as quais realizam seleção dinâmica de *ensembles*. O DCS_LAW_3 obteve o melhor resultado entre os métodos deste grupo para os dados do ISBSG, no entanto, foi inferior aos modelos PEETACO-DES. A Tabela 10 apresenta a superioridade do PEETACO-DES 5C, em uma comparação usando o teste de *T-Student*.

Na DS simples aplicada aos dados do PROMISE, novamente o DCS_LAW, de maneira geral, foi o melhor entre os métodos deste grupo, conforme apresentado na Figura 41. Ao considerar diferentes métricas de avaliação, o DCS_LAW_7 confirmou a sua superioridade sobre os demais métodos de seleção dinâmica mas, dessa vez, foi ultrapassado pelo PEETACO-DS na maioria das métricas avaliadas, como é mostrado na Tabela 20. Em contrapartida, porém, no PROMISE, os métodos de DS simples, incluindo o DCS_LAW, obtiveram um desempenho abaixo do alcançado pelo melhor *ensemble* estático concorrente, e foi superado significativamente pelo PEETACO-DES. Nos dados educacionais, os métodos de DS superaram a seleção estática individual e os *ensembles* homogêneos, apenas.

A partir dos resultados apresentados, é possível perceber, de forma geral, que existe uma superioridade dos métodos propostos nesta tese em relação aos métodos de seleção dinâmica concorrentes (NUCCI, DCS-LA, DCS-LAW). É possível que o principal motivo desta melhoria seja a seleção de *ensembles*, utilizando diferentes critérios, além do uso de diferentes classificadores, visto que essas são as principais diferenças na implementação dos métodos.

No contexto desta pesquisa, os métodos de DS simples, que são comumente usados na literatura, não conseguiram melhorar a precisão alcançada pelas combinações dinâmicas heterogêneas que foram propostas. Em contraste, O PEETACO-DS, de maneira geral, foi superior aos métodos de DS simples em algumas bases de dados. O PEETACO-DS foi superior aos demais métodos de DS simples em quatro das oito bases de dados do PROMISE, no repositório do ISBSG, e em 50% dos conjuntos educacionais. Portanto, acreditamos que o uso de um classificador específico para o conjunto de dados - a fim

de realizar a seleção dinâmica de um modelo de regressão em um *ensemble* heterogêneo - tende a melhorar a precisão das estimativas alcançadas por outros métodos de DS simples, a depender do conjunto de dados avaliado.

Por fim, ao comparar os métodos de DES com os métodos propostos no contexto do ISBSG, percebe-se novamente a superioridade do PEETACO, como apresentado nas Tabelas 10 e 11. Quanto às versões do KNORA, o KNORA_U foi melhor que o KNORA_E, sendo a versão com *k=3* a que obteve melhor resultado. No entanto, conforme apresentado nas Figuras 35 e 36, ele foi superado por várias versões do PEETACO-DES, em especial pela versão PEETACO-DES 5C.

Analisando os resultados nas bases de dados do ISBSG e do PROMISE, percebe-se que o KNORA_U confirma a sua superioridade em relação ao KNORA_E, muito embora o método tenha vencido o PEETACO-DES em apenas uma das oito bases de dados. De maneira geral, considerando todas as bases de dados, o PEETACO-DES, em especial o PEETACO-DES 4C e PEETACO-DES 5C, superou significativamente todas as versões do KNORA, conforme apresentado na Figura 42. A Tabela 20 confirma essa superioridade ao considerar diferentes métricas. Nos dados educacionais, o uso de DES obteve um bom desempenho, quando considerado os métodos gerados pelo *framework* proposto. Os demais métodos de DES foram inferior aos *ensembles* heterogêneos (STACKING e MÉDIA). O KNORA_U foi o método de melhor desempenho entre os de seleção dinâmica, sendo superado apenas pelo PEETACO-DES.

A superioridade alcançada pelos métodos propostos sob os métodos de DES pode ser justificada pelo uso de diferentes critérios de seleção. O KNORA seleciona muitos modelos, mas usando apenas um critério. Além disso, o KNORA é um método original para selecionar classificadores, não regressores, conforme explicado no Item 2.2.5. Se considerarmos os resultados apresentados, o KNORA_U esteve entre os melhores métodos concorrentes, tanto para os dados do ISBSG, quanto para os dados do PROMISE e os educacionais, o que faz com que esse fato favorece o uso de DES. Assim, acreditamos que os resultados alcançados com o uso do KNORA se devem à sua capacidade de combinar métodos individuais de forma dinâmica.

Ainda é possível notar que o PEETACO-DES e o PEETACO-DS não apresentaram desempenhos semelhantes, ocorrência que provavelmente se justifica devido a não utilização de *ensembles* dinâmicos no método PEETACO-DS. Em quase todas as bases de dados, as versões do PEETACO-DES foram melhores do que o PEETACO-DS, que aplica apenas a técnica de seleção dinâmica simples, sem a utilização de fusão, fato esse que pode ser confirmado através dos testes estatísticos realizados. Em todos eles, as versões do PEETACO-DES com três, quatro e cinco classificadores foram superiores ao PEETACO-DS, sendo que apenas a versão com dois classificadores não superou significativamente o PEETACO-DS em todos os testes, apesar desta versão ter tido melhor desempenho em todas as bases de dados. É razoável afirmar que essas poucas semelhanças com PEETACO-

DES 2C foram em virtude do PEETACO-DS possuir características fundamentais que são semelhantes às versões do PEETACO-DES. Exemplo disso é o uso de seleção dinâmica e de classificadores para selecionar os regressores heterogêneos.

Com base nesses resultados, pode-se perceber que os métodos propostos superam todos os modelos concorrentes. As diferentes formas de combinação utilizadas no PEETACO-DES superaram os modelos individuais, suas combinações heterogêneas, homogêneas, os métodos de DS simples e os métodos de seleção dinâmica de múltiplos modelos.

Como se pode perceber, a análise descrita anteriormente focou nos resultados alcançados pelos métodos gerados a partir do *framework* proposto, no entanto, a fim de fortalecer os resultados apresentados, outros pontos da pesquisa ainda podem ser discutidos. Por exemplo: (i) quanto ao uso dos classificadores para seleção dinâmica; e (ii) quanto à possibilidade de existirem tipos de bases de dados adequados ao *framework* proposto. Estas abordagens serão discutidas nas questões de pesquisa 04 e 05.

6.4.4 Questão de pesquisa 04

Existe algum critério de seleção dinâmica capaz de sobressair, quanto à capacidade de selecionar regressores heterogêneos?

O objetivo desta questão de pesquisa é avaliar os classificadores mais adequados para o contexto do *framework* proposto. Nesse sentido, foi estabelecida uma pontuação para cada classificador utilizado nos conjuntos de dados. Essa pontuação foi baseada em um ranqueamento inverso, sendo 5 pontos para o algoritmo selecionado como primeiro classificador e, 4, 3, 2 e 1 ponto para os demais algoritmos, respectivamente. Após esta pontuação os algoritmos foram classificados conforme a Tabela 26.

A partir do resultado apresentado na Tabela 26, percebe-se que os algoritmos *Adaboost*(FREUND; SCHAPIRE, 1996) e *KNN* somaram as maiores pontuações e, além disso, estiveram presentes na maioria das bases de dados de EES. Com esse resultado, podemos reforçar o poder da seleção dinâmica por localidade, visto que o *KNN* é um algoritmo baseado em distância. Além do mais, o uso de árvores de decisão para tal finalidade também apresentou excelentes resultados, uma vez que o *Adaboost* utilizado foi definido com árvores de decisão em seu classificador base, conforme apresentado no código fonte do Apêndice B. Nesse sentido, os algoritmos baseados em árvores de decisão atuam com relevância na seleção dinâmica de modelos de regressão heterogêneos. Somando ao bom desempenho das árvores de decisão, podemos destacar os algoritmos: (i) *J48*, que foi o classificador base mais usado no *Adaboost* - sendo em 4 das 6 vezes - e (ii) o algoritmo *Rep Tree*, o qual foi usado em 6 das 9 bases de dados e ficou em terceiro lugar no ranqueamento geral. Assim, algoritmos baseados em distância e em árvores de decisão destacam-se quanto à seleção dinâmica proposta nesta pesquisa, porém, ainda não é possível afirmar que individualmente eles superem os demais algoritmos.

Tabela 26 – Pontuação dos algoritmos de classificação usados como seletores de modelos de regressão em EES

Algoritmos	ISBSG	China	Coc. 81	Coc. Nasa V1	Coc. Nasa V2	Desh.	Kitch.	Maxwell	Myiaz.	Soma
AD		4	5			5	4	5	2	25
KNN	5		4	1	1		5	4	5	25
RT		2	1	2		2		2	4	13
BA		5		3		1	3			12
KS	3	1	3			4		1		12
RF	1	3	2			3		3		12
DT				5	5				1	11
SVM					3				3	6
J48	4						1			5
BFT				4						4
RBF					4					4
MLP	2									2
LT					2					2
LWL							2			2

Fonte: O autor (2022)

A Seção 6.3 apresentou os resultados da aplicação de vários métodos de estimativa no domínio de dados educacionais. Com o fito de investigar se o que foi abordado no parágrafo anterior pode se repetir em outros contextos, foi realizado um ranqueamento semelhante nos seis conjuntos de dados educacionais, cujo resultado mostrou que o *KNN* venceu novamente, desta vez seguido do algoritmo *Random Forest*, mesmo que a diferença de pontuação entre os melhores classificadores tenha sido menor do que a apresentada na análise anterior, como pode ser visto na Tabela 27 que apresenta o resultado desta pontuação, o que não se repetiu na mesma proporção para o *Adaboost*. Nessa avaliação os melhores classificadores foram mais diversos, principalmente com o uso de algoritmos baseados em funções estimadas, como, por exemplo, *Support Vector Machine* (SVM) e *Multi Layer Perceptron* (MLP). As possíveis justificativas para estes resultados são as mudanças de domínios, que levaram as bases de dados a terem características distintas. Desta forma, os tipos de dados e a distribuição dos valores das variáveis foram analisados e, são características que podem ser estudadas com mais detalhes no futuro. Vale destacar também que é possível que o CS utilizado em cada base de dados seja influenciado pelo CB selecionado anteriormente. Por fim, as variáveis dependentes dos dados educacionais são em sua maioria numéricas, o que pode facilitar ou dificultar o aprendizado de alguns algoritmos. Por exemplo, as MLP conseguiram gerar melhores seletores dinâmicos, quando comparado aos dados de EES. Apesar de nenhum classificador se destacar nesta análise, perceber-se que os algoritmos baseados em distância e as árvores de decisão con-

tinuaram sendo boas opções para seleção dinâmica de regressores heterogêneos, o que se pode concluir, consequentemente, que estes resultados poderão levar os estudos futuros a utilizarem apenas esses tipos de algoritmos como estratégia inicial de seleção dinâmica na utilização do *framework* proposto.

Tabela 27 – Pontuação dos algoritmos de classificação usados como seletores de modelos de regressão nas bases de dados educacionais

Algoritmos	TAF	TAM	TEF	TEM	TRF	TRM	SOMA
KNN			1	5	4	4	14
RF			3	4	5		12
BA	1	4	4	2			11
MLP		5			3	2	10
AD		2	5	1			8
LoR	5						5
NB	4					1	5
LMT	3				2		5
SVM	2	3					5
KS						5	5
LWL				3			3
RT						3	3
J48			2				2
DS		1					1
RBF					1		1

Fonte: O autor (2022)

6.4.5 Questão de pesquisa 05

Existem meta-características nas bases de dados ou entre os modelos individuais que favoreçam o uso do *framework* proposto?

A revisão sistemática realizada por Wen et al. (WEN et al., 2012) afirma que o desempenho de qualquer modelo depende principalmente das características dos conjuntos de dados usados para construir o modelo (tamanho do conjunto de dados, características categóricas discrepantes, valores ausentes etc.). A partir desta afirmação, torna-se importante investigar a relação de diferentes características dos dados com o *framework* proposto.

De maneira geral, os métodos oriundos do *framework* proposto foram superiores aos métodos concorrentes no contexto dos dados de EES e nos dados educacionais. No entanto, em algumas bases de dados, a exemplo da DESHARNAIS, o PEETACO não obteve um bom resultado. Esta abordagem de identificar tipos de bases de dados não adequadas ao uso do *framework* proposto será analisada a seguir.

As características das bases de dados, assim como as estatísticas das variáveis dependentes foram investigadas não apenas quanto aos regressores e classificadores adequados, conforme mencionado, mas também quanto ao uso do próprio *framework*. As Tabelas 5, 6, 7, 148, 22 trazem características das bases de dados, das variáveis dependentes e dos resultados alcançados pelos regressores individuais na fase de validação. Os valores coletados foram usados com o objetivo de descobrir características relacionadas ao desempenho do PEETACO, sendo que no total, 19 características foram investigadas e, cada base de dados foi rotulada quanto ao desempenho do PEETACO-DES 5C. A primeira análise foi referente à posição do método na base de dados, sendo essa informação definida em uma variável qualitativa ordinal. A segunda análise foi definida em uma variável binária em que, para cada base de dados, foi indicado se o PEETACO foi vencedor. De maneira geral, o PEETACO venceu em 11 das 15 bases de dados investigadas, porém, a fim de deixar essa análise de dados mais balanceada, o PEETACO só foi considerado vencedor, quando o método PEETACO-DES 5C foi superior a todos os concorrentes.

Após análises utilizando os testes estatísticos adequados, duas meta-características das bases de dados apresentaram possíveis correlações com o desempenho dos métodos propostos, são elas: (i) o número de instâncias na base de dados e (ii) a quantidade de atributos numéricos. A Tabela 28 apresenta esses metadados juntamente com o desempenho do PEETACO-DES 5C em cada base de dados. A coluna *ranking* refere-se a posição geral do PEETACO-DES 5C, inclusive considerando as demais configurações do PEETACO que foram avaliadas. Quanto à relação desses atributos com a variável binária que indica se o método foi vencedor, os dados foram divididos em dois grupos: (i) o grupo rotulado com o valor *Sim* para a variável *Venceu* e (ii) o grupo rotulado com o valor *Não* para indicar que o PEETACO-DES 5C perdeu para um dos concorrentes. Desta forma, foi considerado vencedor, apenas quando ele ficou a frente de todos os métodos concorrentes.

Uma vez que os valores da quantidade de atributos numéricos e do número de instâncias não obedeceram uma distribuição normal, e que o *Ranking* é uma variável ordinal, foi usado o coeficiente de correlação de *Spearman* para verificar a relação entre essas variáveis. Além disso, visto que os grupos de dados *Venceu=Sim* e *Venceu=Não* são independentes e não paramétricos, o teste de *Mann-Whitney U* foi usado para verificar se existia diferença significativa entre os valores da amostra.

A Figura 46 apresenta os resultados da correlação de *Spearman* entre as variáveis *Quantidade de Atributos Numéricos* (Qtd. Num.) e *Instâncias* com o *Ranking* do PEETACO-DES 5C. O resultado do teste de *Mann-Whitney* aplicado aos grupos *Venceu = Sim* e *Venceu = Não* em relação à quantidade de instâncias de cada base de dados é apresentado em seguida na Figura 47.

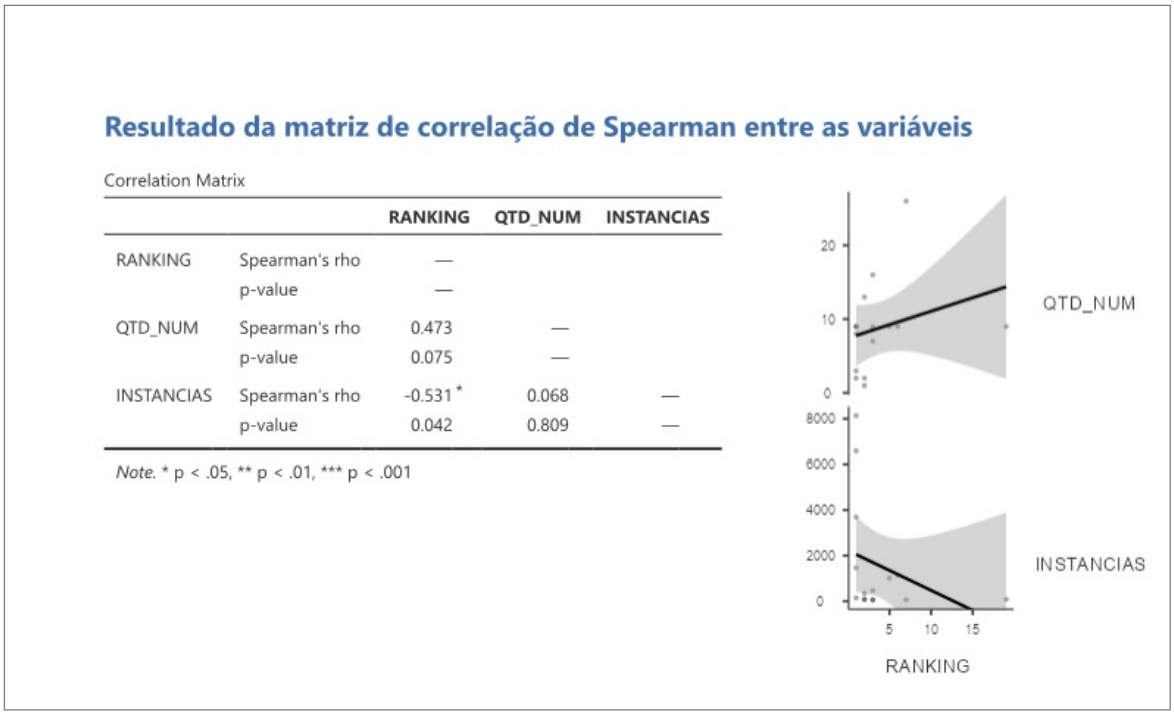
Em relação a quantidade de atributos numéricos, é possível perceber a partir do resultado apresentado na Figura 46 que existe uma correlação moderada e positiva com o *Ranking* do PEETACO-DES 5C. Consequentemente, isto implica que o *Ranking* do

Tabela 28 – Desempenho do PEETACO-DES 5C por base de dados

Base de dados	**Qtd. Num.**	**Instâncias**	**Ranking**	**Venceu**
ISBSG	3	1466	1	Sim
China	13	346	2	Sim
Cocomo81	16	63	3	Não
Cocomonasa V1	1	60	2	Não
Cocomonasa V2	2	93	2	Sim
Desharnais	9	81	19	Não
Kitchenham	2	145	1	Sim
Maxwell	26	62	7	Não
Myiazaki	7	48	3	Não
Aprovação Fundamental	9	8132	1	Sim
Aprovação Médio	9	1135	6	Não
Evasão Fundamental	8	3683	1	Sim
Evasão Médio	9	468	3	Sim
Reprovação Fundamental	9	6592	1	Sim
Reprovação Médio	9	1012	5	Não

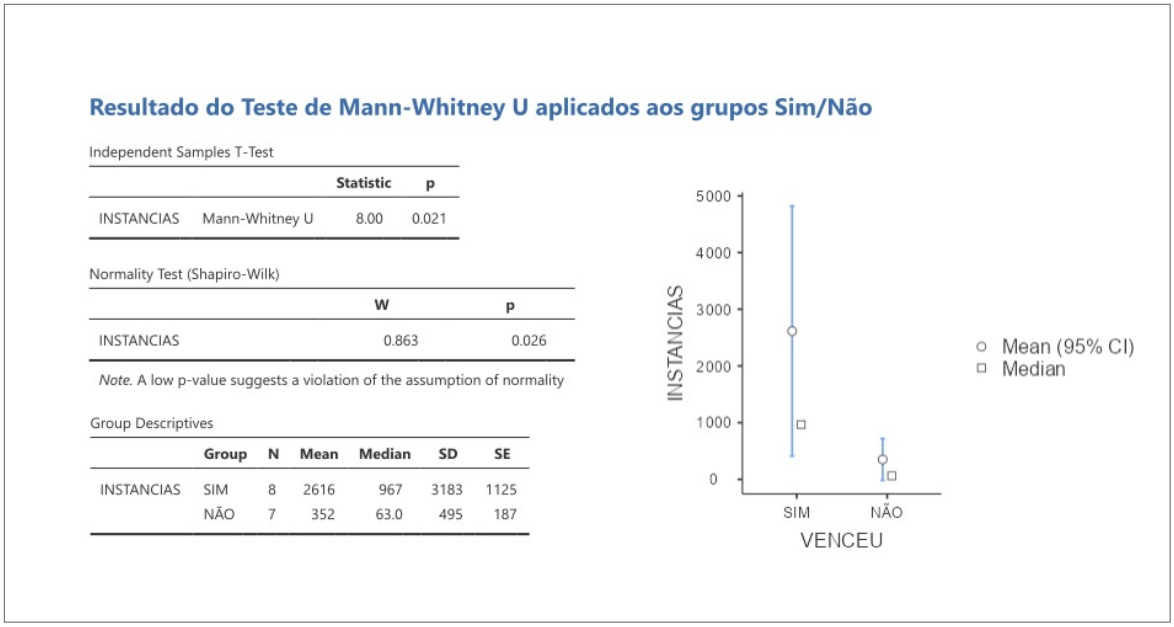
Fonte: O autor (2022)

Figura 46 – Resultado estatístico do teste de correlação de *Spearman* aplicado às características *Quantidade de Atributos Numéricos* e *Número de Instâncias* com o *Ranking* do PEETACO-DES 5C



Fonte: O autor (2022)

Figura 47 – Resultado estatístico do teste de *Mann-Whitney U* aplicado aos grupos *Sim* e *Não* da variável *Venceu*, quanto ao método PEETACO-DES 5C



Resultado do Teste de Mann-Whitney U aplicados aos grupos Sim/Não

Independent Samples T-Test

		Statistic	p
INSTANCIAS	Mann-Whitney U	8.00	0.021

Normality Test (Shapiro-Wilk)

	W	p
INSTANCIAS	0.863	0.026

Note. A low p-value suggests a violation of the assumption of normality

Group Descriptives

	Group	N	Mean	Median	SD	SE
INSTANCIAS	SIM	8	2616	967	3183	1125
	NÃO	7	352	63.0	495	187

Fonte: O autor (2022)

método possibilita uma tendência a piorar com o aumento de atributos numéricos. No entanto, esta correlação deve ser investigada com mais detalhes em estudos futuros, uma vez que o valor *p-value > 0,05* indica que o valor da correlação não foi de fato significativo, mas como o *p-value* foi próximo de *0,05* (*p-value = 0,075*), podemos dizer que a diferença foi marginalmente significativa. Neste sentido, sugere-se aumentar o tamanho da amostra e monitorar o *p-value*, a fim de alcançar um resultado mais confiável, uma vez que o poder do teste também será maior. Esta possível correlação entre as duas variáveis pode ser explicada pelos tipos de classificadores que obtiveram melhores desempenhos no contexto deste trabalho. Por exemplo, o KNN é um algoritmo que funciona melhor em bases de dados com dimensões menores, enquanto os algoritmos SVM e MLP, os quais conseguem ter melhores desempenhos em dimensões maiores, foram pouco usados como seletores no contexto deste trabalho.

Quanto à influência do *Número de Instância* no desempenho do método, há uma correlação aparentemente mais forte do que a anterior, mas com um direcionamento negativo. Nesta análise, com $r > 0,5$ e o $p - value < 0,05$ temos uma correlação de moderada para forte e significativa, conforme apresenta a Figura 46. No entanto, o teste *r-to-z* de transformação de *Fisher* demonstrou que o desempenho do método em relação ao *ranking* obtido não se associou mais fortemente com o número de instâncias (*r=-0,531, p-value = 0,042)* do que com a quantidade de atributos numéricos (*r=0,473, p-value=0,075*) (*z=-0,185,p-value=0,426*). Além disso, o coeficiente de determinação calculado a partir do resultado de r, mostrou que o número de instâncias explica cerca de 28% da variação dos *Rankings* alcançados pelo PEETACO-DES 5C. No entanto, a influência da quanti-

dade de instâncias no desempenho do método pode ser fortalecida através do resultado do teste de *Mann-Whitney U* apresentado na Figura 47. O resultado demonstrou que existe uma diferença significativa entre os valores do *Número de Instâncias* das bases de dados com relação ao desempenho do método (Venceu = Sim/Não). O grupo de base de dados onde o PEETACO-DES 5C *Venceu = Sim* se difere significativamente do grupo *Venceu = Não* em relação ao número de instâncias. Estes resultados indicam que o método tende a ter melhores desempenhos com bases de dados maiores, e que este fenômeno pode ser explicado pelo tamanho da Base de Treinamento dos Classificadores (τ_c). Bases de dados com muitas instâncias geram τ_c maiores e, consequentemente, os classificadores do CS apontam para a possibilidade de gerar modelos mais generalizáveis, o que levará os métodos a alcançarem melhores resultados. Apesar das correlações apresentadas, ainda é difícil garantir que elas sejam realmente verdadeiras, já que o tamanho das amostras ainda é relativamente pequeno. Estudos futuros serão necessários para confirmar essas descobertas.

6.4.6 Ameaças à validade

Conduzimos quatro tipos de ameaças à validade. Uma abordagem para cada um desses tipos será apresentada abaixo.

Validade de conclusão: este tipo de ameaça verifica se há relação entre o fator que foi alterado (o tratamento) e os resultados que obtemos (as respostas). Nesta tese, os métodos avaliados são os tratamentos, enquanto os erros das estimativas de cada método são as respostas, assim, os dados coletados poderiam levar ou não a uma relação entre os fatores e as respostas. Com o objetivo de suprimir as principais ameaças às conclusões, especialmente em termos de resultados estatísticos, os experimentos foram realizados a um nível de significância de 5% em 15 bases de dados distintas e, a um poder de teste de 80%. Desta forma, para todos os experimentos, foi realizado um cálculo amostral para um tamanho de efeito alto e para o teste estatístico adequado, com $\alpha = 0,05\%$ e $\beta = 80\%$. Os testes estatísticos utilizados em cada experimento foram adequados ao problema em questão. Neste sentido, foi verificado a distribuição dos dados, a existência de paridade entre as amostras e a quantidade de grupos amostrais. A fim de evitar que os testes fossem realizados em dados que sempre levassem ao resultado esperado, foram utilizadas as principais bases de dados de EES disponíveis na literatura e, que foram comumente usadas em outros estudos. Além disso, conjuntos de dados educacionais com características diferentes das bases de EES também foram investigados. Como era de se esperar, os métodos gerados a partir do *framework* proposto não foram superiores aos demais métodos em todas as bases de dados, já que, por melhor que seja o método, é comum que ele não seja superior aos demais em todos os domínios de dados, no entanto, os métodos oriundos do *framework* proposto venceram na maioria das bases de dados. Outra estratégia utilizada para evitar falhas de conclusão, foi a utilização de testes *post-hoc*, os

quais evitam erros cumulativos, quando a conclusão final depende do resultado de mais de um experimento. Além dos testes estatísticos realizados, a partir dos erros absolutos de cada métodos, outras cinco métricas foram adicionadas para fortalecer a conclusão dos resultados. Quanto à implementação dos métodos, foi disponibilizado o código fonte de cada algoritmo, o que permite que eles sejam verificados e validados por qualquer usuário. Por fim, como os tratamentos foram representados pelos próprios algoritmos, os experimentos não sofreram alterações randômicas que pudessem afetar os resultados, além disso, as unidades experimentais permaneceram fixas para todos os métodos avaliados, a fim de evitar que as possíveis diferenças entre os fatores viessem a ocorrer pela mudança das unidades experimentais.

Validade interna: na engenharia de software, essa validade pode ser muito afetada pelo comportamento dos sujeitos que participam dos experimentos. No entanto, neste trabalho, os tratamentos e os grupos não estão relacionados as pessoas, mas sim aos algoritmos e aos dados de projetos de *software*. A qualidade da coleta dos dados pode ser uma ameaça à validade interna, porém, nos dados do ISBSG apenas os projetos com nível de qualidade A ou B do repositório foram selecionados. Em relação às bases de dados do PROMISE, estudos recentes têm mostrado que elas são amplamente utilizadas, quando se trata de problemas de EES (IDRI; HOSNI; ABRAN, 2016b; IDRI; HOSNI; ABRAN, 2016a) e assim possuem solidez nesta área. Os dados educacionais foram usados a partir de uma coleta de alta qualidade realizada a partir de informações públicas (NASCIMENTO; FAGUNDES; MACIEL, 2019). Quanto aos algoritmos implementados, cada um tem implementações específicas, onde a ameaça de imitação do tratamento não se aplica. Desta forma os tratamentos foram equalizados, sendo colocados sob as mesmas condições. De maneira geral, as ameaças internas não afetam os resultados apresentados neste trabalho.

Validade de constructo: este tipo de ameaça investiga se a causa e efeito dos resultados estão relacionados com a hipótese levantada e o que foi observado. No Capítulo 4, apresentamos as principais características específicas do *framework* proposto que podem levar a melhorias, quanto ao desempenho de diversos métodos. Podemos citar como exemplo, o uso de *ensembles* de classificadores e regressores heterogêneos, de um processo fusão, da estratégia de seleção dinâmica de modelos, do uso de diferentes critérios de seleção etc. Além disso, conjuntos de dados com naturezas e atributos distintos e com diferentes características foram usados, a fim de representar a população de dois domínios que têm apresentado o uso de *ensembles* em seus estudos. Há dois vieses que podem afetar o design do experimento, um é o viés mono-operação, relacionado as variáveis independentes; e o outro, o mono-método, relacionado a variável dependente, por exemplo, essa ameaça pode está associada a quantidade de níveis usadas no fator observado na variável independente, porém, neste trabalho, foram investigados vários métodos de estimativa nos experimentos realizados, e, conforme explicado, cada método foi abordado como um tratamento do experimento, e que nesse contexto, a mudança de tratamentos é

semelhante a variação de níveis do fator. Consequentemente, o fator método considerou vários níveis, sendo que, a quantidade de níveis testadas está acima do que é realizado nas comparações de estudos anteriores. Neste trabalho, propomos um *framework* que utiliza classificadores para selecionar modelos de regressão que compõem um *ensemble* heterogêneo, no entanto, seria possível questionar se os resultados permaneceriam os mesmos, se usássemos diferentes quantidades de classificadores. Assim, no Capítulo 6 os experimentos relatados apresentaram os resultados usando 1, 2, 3, 4 e 5 classificadores com o propósito de investigar se o desempenho do método seria alterado. Os resultados apresentados mostraram que o *ensemble* dinâmico tende a melhorar o desempenho com o aumento do número de classificadores, mas que, posteriormente, o desempenho manteve-se constante ao atingir o tamanho adequado. Outra ameaça relacionada ao constructo, seria quanto a melhoria em um aspecto provocar a piora em outro. Nos experimentos realizados neste trabalho, o *framework* proposto conseguiu melhorar as estimativas dos métodos individuais e combinados, entretanto, para alguns contextos poderá se tornar inadequado se o tempo de resposta necessário for real. Por fim, A interação entre os testes e os tratamentos e as ameaças sociais à validade de constructo poderiam afetar os experimentos, entretanto, por se tratar de algoritmos, os testes em si não interage com os tratamentos nem ameaçam a validade de constructo socialmente.

Validade externa: este tipo de ameaça investiga se é possível generalizar os resultados em outras áreas. A generalização dos métodos propostos foi melhor do que a dos demais encontrados no estado da arte, portanto, acreditamos que podemos obter avanços em relação ao desempenho do PEETACO em outros contextos, uma vez que os resultados alcançados, tanto no domínio de EES, quanto no de EDM, foram relativamente superiores à maioria dos métodos concorrentes. Em hipótese, os resultados poderão ser generalizados nestas áreas e, quanto aos demais domínios existentes, o *framework* poderá ser facilmente replicado e investigado para problemas de regressão. Porém, vale ressaltar que, os métodos sugeridos não são adequados para contextos que requerem alto desempenho em termos de tempo de processamento, além disso, o desempenho dele não tende a melhorar em bases de dados relativamente pequenas, ou se tiverem muitas variáveis independentes.

7 CONCLUSÃO

Esta seção tem como objetivo apresentar as considerações finais sobre os principais tópicos abordados nesta tese, incluindo as contribuições alcançadas e indicações para trabalhos futuros.

7.1 CONSIDERAÇÕES FINAIS

Este trabalho apresentou um novo *framework* de seleção dinâmica de *ensembles* heterogêneos para problemas de regressão. O *framework* proposto foi avaliado em problemas de EES e em bases de dados educacionais. O método é baseado na ideia de utilizar classificadores treinados para selecionar os melhores regressores a serem usados para fornecer a estimativa para uma dada entrada. A saída de cada modelo de regressão é ponderada dinamicamente, de acordo com a saída dos classificadores usados.

No procedimento, foi incluída uma fase de validação para selecionar os melhores regressores (CB) e classificadores (CS). O uso do *framework* sugerido é sólido, e a superioridade alcançada em comparação aos demais métodos foi confirmada pela aplicação de testes estatísticos adequados e aplicados em 15 conjuntos de dados a um nível de confiança de com 95 %.

A seleção dinâmica foi construída levando em consideração o desempenho e a diversidade dos modelos. Além disso, classificadores foram avaliados, a fim de selecionar os melhores regressores em cada instância para cada conjunto de dados. O *framework* proposto é único e os métodos construídos a partir dele (PEETACO) superaram os do estado da arte.

O PEETACO alcançou avanços significativos em diferentes bases de dados e, por essa razão, conduz a ter bons resultados em outros domínios, visto o desempenho que também foi obtido nas bases de dados educacionais. Acreditamos que utilizar classificadores de naturezas diferentes no CS, a exemplo dos métodos baseados em distância e em árvores, juntamente com a integração dos regressores do CB selecionados dinamicamente, tornou-se fundamental para termos resultados bem sucedidos.

Neste sentido, o uso de classificadores para selecionar dinamicamente modelos de regressão heterogêneos parece ser uma boa estratégia a ser usada em sistemas de múltiplos modelos e, deve continuar sendo avaliada em novos estudos. Métodos tradicionais de seleção dinâmica, em geral, foram semelhantes aos métodos de seleção de estática de ensembles, como, por exemplo, o desempenho do *ensemble* heterogêneo (mediana) que alcançou resultados semelhantes ou com pequenas melhorias em relação a alguns métodos de seleção dinâmica conhecido na literatura e adaptados ao contexto deste trabalho. No entanto, a proposta de usar diferentes critérios de seleção dinâmica através de classifica-

dores adequados a cada base de dados para ponderar as saídas dos regressores superou, não apenas métodos de seleção dinâmica, mas também os *ensembles* estáticos.

Por fim, o *framework* proposto utilizou diferentes classificadores para selecionar modelos de regressão e, dentre eles, foram usados modelos de diferentes naturezas. A busca pelos melhores modelos foi fundamental para melhorar a precisão da seleção dinâmica de *ensembles*, enquanto a diversidade de critérios de seleção foi fundamental para o alcance de resultados significativos. Os algoritmos baseados em distância e as árvores de decisão foram os grupos de classificadores mais estáveis entre todos avaliados. Isto indica que a seleção dinâmica baseada nesses critérios, de maneira geral, apresentou-se mais adequada para selecionar modelos de regressão em um *ensemble* heterogêneo. Ainda, a partir das análises realizadas, foi mostrado que existem indícios de que, quanto maior for o número de instâncias na base de dados, melhor tende a ser o desempenho dos métodos gerados a partir do *framework* proposto, contrariamente, a alta dimensionalidade dos dados pode afetar o desempenho do método.

7.2 PRINCIPAIS CONTRIBUIÇÕES E TRABALHOS FUTUROS

A fim de garantir qualidade nos resultados presentes, este trabalho apresenta: (i) objetivos claros e bem definidos; (ii) uma solução apresentada e discutidas no Capítulo 6; (iii) aplicação dos métodos avaliados em 15 diferentes bases de dados; (iv) desempenho medido em seis diferentes tipos de métricas; (v) comparação dos métodos propostos com os modelos individuais e suas combinações homogêneas e heterogêneas, além de diversas estratégias de seleção dinâmica. Ainda, até o presente momento, dois estudos foram publicados, sendo o primeiro em conferência internacional (CABRAL et al., 2017) e a segundo em periódico (CABRAL; OLIVEIRA, 2021).

Para dar continuidade ao estudo desta tese, listam-se, nesta seção, possíveis propostas de trabalhos futuros a serem realizadas.

1. Investigar o *framework* proposto em outros conjuntos de dados de EES; esta é a sugestão inicial de trabalhos futuros, novas bases de dados surgirão na literatura e os modelos de AM são promissores; nesse sentido, as empresas podem se beneficiar deste *framework* proposto a partir de seus próprios dados históricos. A tarefa inicial seria encontrar o CB e o CS adequado ao conjunto de dados. Entretanto, a implementação de AM ainda é limitada na indústria e, dessa forma, algumas organizações poderiam iniciar com os conjuntos de dados usados neste trabalho.

2. Replicar os experimentos realizados em dados aleatórios, a fim de avaliar o desempenho do PEETACO de maneira geral. Esta avaliação seria uma forma de dar maior robustez ao uso do *framework*. A avaliação de desempenho pode vir a partir do desempenho dos métodos gerados por ele em diferentes bases de dados. Esta é

uma alternativa para conhecer os tipos de conjuntos de dados mais adequados ao *framework*.

3. Aplicar a seleção de atributos na fase de validação para investigar o desempenho do *framework* proposto; outra abordagem que tem sido utilizada para se chegar a resultados mais precisos em AM é a seleção de atributos (*feature selection*). Este procedimento pode melhorar a precisão dos modelos individuais e, portanto, a precisão dos modelos combinados (HOSNI et al., 2017).

4. Associar os classificadores adequados as bases de dados; além de selecionar os melhores recursos, também é possível selecionar os parâmetros dos classificadores usados dentro do *framework* proposto, objetivando melhorar o desempenho na seleção dinâmica. Esta melhoria poderá levar a resultados ainda mais satisfatórios.

5. Utilizar diferentes funções para fusão dos modelos selecionados dinamicamente; os métodos de fusão que foram usados neste trabalho são lineares. Outra sugestão de investigação futura seria avaliar diferentes estratégias que fossem capazes de combinar os regressores de maneiras distintas das apresentadas neste trabalho, por exemplo, uma regressão não linear, que poderá melhorar o desempenho do *framework* proposto na fase de integração.

6. Investigar o *framework* proposto em ensembles homogêneos; todas as avaliações realizadas consideraram apenas *ensembles* heterogêneos. No entanto, sabemos que os *ensembles* homogêneos também são capazes de melhorar o desempenho dos modelos bases. Desta maneira, adaptar o *framework* proposto à seleção dinâmica de múltiplos modelos baseados no mesmo algoritmo e investigar o resultado desta adaptação é uma possibilidade de avanço do próprio SMM.

REFERÊNCIAS

ABDELALI, Z.; MUSTAPHA, H.; ABDELWAHED, N. Investigating the use of random forest in software effort estimation. *Procedia Computer Science*, v. 148, p. 343–352, 2019. ISSN 1877-0509. THE SECOND INTERNATIONAL CONFERENCE ON INTELLIGENT COMPUTING IN DATA SCIENCES, ICDS2018. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1877050919300420>.

ABNANE, I.; HOSNI, M.; IDRI, A.; ABRAN, A. Analogy software effort estimation using ensemble knn imputation. In: *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. [S.l.: s.n.], 2019. p. 228–235.

ABRAN, A. Data collection and industry standards: The isbsg repository. In: *Software Project Estimation*. [S.l.]: John Wiley and Sons, Inc, 2015. p. 161–184. ISBN 9781118959312.

AHA, D.; KIBLER, D.; ALBERT, M. Instance-based learning algorithms. *Mach Learn*, v. 6, p. 37–66, Jan. 1991.

AMARAL, W.; RIVERO, L.; JUNIOR, G. B.; VIANA, D. Using machine learning technique for effort estimation in software development. In: . New York, NY, USA: Association for Computing Machinery, 2019. (SBQS'19), p. 240–245. ISBN 9781450372824. Disponível em: <https://doi.org/10.1145/3364641.3364670>.

AMASAKI, S. A comparative study on linear combination rules for ensemble effort estimation. In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. [S.l.: s.n.], 2017. p. 104–107.

ANOZIE, N. O.; JUNKER, B. W. Predicting end-of-year accountability assessment scores from monthly student records in an online tutoring system. In: . [S.l.: s.n.], 2006.

ARAUJO, R. de A.; OLIVEIRA, A. L.; SOARES, S.; MEIRA, S. An evolutionary morphological approach for software development cost estimation. *Neural Networks*, v. 32, n. Supplement C, p. 285 – 291, 2012. ISSN 0893-6080.

ARAUJO, R. de A.; SOARES, S.; OLIVEIRA, A. L. Hybrid morphological methodology for software development cost estimation. *Expert Systems with Applications*, v. 39, n. 6, p. 6129 – 6139, 2012. ISSN 0957-4174.

ARGAWAL, R.; KUMAR, Y.; MALLICK, Y.; BHARADWAJ, R.; ANANTWAR, D. Estimating software projects. *Software Engineering*, v. 26, n. 4, p. 60–67, 2001.

ARNOLD, A.; SCHEINES, R.; BECK, J.; JEROME, B. Time and attention: Students, sessions, and tasks. 01 2005.

ATKESON, C. G.; MOORE, A. W.; SCHAAL, S. Locally weighted learning. *Artif. Intell. Rev.*, Kluwer Academic Publishers, USA, v. 11, n. 1–5, p. 11–73, feb 1997. ISSN 0269-2821. Disponível em: <https://doi.org/10.1023/A:1006559212014>.

AYERS, E.; JUNKER, B. Do skills combine additively to predict task difficulty in eighth grade mathematics? In: *In AAAI Workshop on Educational Data Mining: Menlo Park*. [S.l.: s.n.], 2006. p. 14–20.

AZEVEDO, L. P. *National Institute of Educational Studies and Research Anísio Teixeira*. 2018. <http://portalinep.gov.br/>. [Online; acessado em 14 dez 2018].

AZZEH, M.; NASSIF, A. B.; BANITAAN, S.; LóPEZ-MARTíN, C. Ensemble of learning project productivity in software effort based on use case points. In: *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. [S.l.: s.n.], 2018. p. 1427–1431.

BAKER, E.; MCGAW, B.; PETERSON, P. Data mining for education. n. 1, 2010. Disponível em: <https://learninganalytics.upenn.edu/ryanbaker/Encyclopedia\%20Chapter\%20Draft\%20v10\%20-fw.pdf>.

BAKER, R. S.; YACEF, K. The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining*, v. 1, n. 1, p. 3–17, Oct. 2009. Disponível em: <https://jedm.educationaldatamining.org/index.php/JEDM/article/view/8>.

BECK, J.; WOOLF, B. P. High-level student modeling with machine learning. In: *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*. Berlin, Heidelberg: Springer-Verlag, 2000. (ITS '00), p. 584–593. ISBN 3540676554.

BEEMER, J.; SPOON, K.; HE, L.; FAN, J.; LEVINE, R. Ensemble learning for estimating individualized treatment effects in student success studies. *International Journal of Artificial Intelligence in Education*, v. 28, 05 2017.

BEEMER, J.; SPOON, K.; HE, L.; FAN, J.; LEVINE, R. Ensemble learning for estimating individualized treatment effects in student success studies. *International Journal of Artificial Intelligence in Education*, v. 28, 05 2017.

BOEHM, B. *Software Engineering Economics.* [S.l.]: Prentice - Hall, 1981.

BOEHM, B.; SULLIVAN, K. Software economics: Status and prospects. In: *Information and Software Technology, Nov 15*. [S.l.: s.n.], 1999. p. 937–946.

BRAGA, A. de P.; CARVALHO, A. de L. F.; LUDERMIR, T. *Redes neurais artificiais: teoria e aplicações*. LTC Editora, 2007. ISBN 9788521615644. Disponível em: <https://books.google.com.br/books?id=R-p1GwAACAAJ>.

BRAGA, P. L.; OLIVEIRA, A. L. I.; RIBEIRO, G. H. T.; MEIRA, S. R. L. Bagging predictors for estimation of software project effort. In: *2007 International Joint Conference on Neural Networks*. [S.l.: s.n.], 2007. p. 1595–1600. ISSN 2161-4393.

BREIMAN, L. Bagging predictors. *Machine Learning*, v. 24, n. 2, p. 123–140, Aug 1996. ISSN 1573-0565.

BREIMAN, L. Random forests. *Machine Learning*, v. 45, p. 5–32, 10 2001.

BRIAND, L. C.; EMAM, K. E.; SURMANN, D.; WIECZOREK, I.; MAXWELL, K. D. An assessment and comparison of common software cost estimation modeling techniques. In: *Proceedings of the 1999 International Conference on Software Engineering (IEEE Cat. No.99CB37002)*. [S.l.: s.n.], 1999. p. 313–323. ISSN 0270-5257.

BRITTO, A. S.; SABOURIN, R.; OLIVEIRA, L. E. Dynamic selection of classifiers—a comprehensive review. *Pattern Recognition*, v. 47, n. 11, p. 3665 – 3680, 2014. ISSN 0031-3203.

BROWN, G.; WYATT, J.; HARRIS, R.; YAO, X. Diversity creation methods: a survey and categorisation. *Information Fusion*, v. 6, n. 1, p. 5 – 20, 2005. ISSN 1566-2535. Diversity in Multiple Classifier Systems.

BROWN, G.; WYATT, J. L.; TINO, P. Managing diversity in regression ensembles. *J. Mach. Learn. Res.*, JMLR.org, v. 6, p. 1621–1650, dez. 2005. ISSN 1532-4435.

CABRAL, J. T. H. de A.; ARAUJO, R. de A.; NOBREGA, J. P.; OLIVEIRA, A. L. I. Heterogeneous ensemble dynamic selection for software development effort estimation. In: *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*. [S.l.: s.n.], 2017. p. 210–217. ISSN 2375-0197.

CABRAL, J. T. H. de A.; OLIVEIRA, A. L. Ensemble effort estimation using dynamic selection. *Journal of Systems and Software*, v. 175, p. 110904, 2021. ISSN 0164-1212. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0164121221000017>.

CABRAL, J. T. H. de A.; OLIVEIRA, A. L.; da Silva, F. Q. Ensemble effort estimation: An updated and extended systematic literature review. *Journal of Systems and Software*, p. 111542, 2022. ISSN 0164-1212. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0164121222002187>.

CARVALHO, H. D. P.; LIMA, M. N. C. A.; SANTOS, W. B.; A.FAGUNDE, R. A. de. Ensemble regression models for software development effort estimation: A comparative study. *International Journal of Software Engineering & Applications*, Academy and Industry Research Collaboration Center (AIRCC), v. 11, n. 3, p. 71–86, May 2020. ISSN 0976-2221. Disponível em: <http://dx.doi.org/10.5121/ijsea.2020.11305>.

CASTRO, F.; VELLIDO, A.; NEBOT, À.; MUGICA, F. Applying data mining techniques to e-learning problems. In: _____. *Evolution of Teaching and Learning Paradigms in Intelligent Environment*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 183–221. ISBN 978-3-540-71974-8. Disponível em: <https://doi.org/10.1007/978-3-540-71974-8_8>.

CAVALIN, P. R.; SABOURIN, R.; SUEN, C. Y. Dynamic selection approaches for multiple classifier systems. *Neural Computing and Applications*, v. 22, n. 3, p. 673–688, Mar 2013. ISSN 1433-3058.

CETINTAS, S.; SI, L.; XIN, Y. P.; HORD, C. Predicting correctness of problem solving from low-level log data in intelligent tutoring systems. In: . [S.l.: s.n.], 2009. p. 230–239.

CHAPMAN, P.; CLINTON, J.; KERBER, R.; KHABAZA, T.; REINARTZ, T. P.; SHEARER, C.; WIRTH, R. Crisp-dm 1.0: Step-by-step data mining guide. In: . [S.l.: s.n.], 2000.

CHARETTE, R. N. Why software fails [software failure]. *IEEE Spectr.*, IEEE Press, Piscataway, NJ, USA, v. 42, n. 9, p. 42–49, set. 2005. ISSN 0018-9235. Disponível em: <http://dx.doi.org/10.1109/MSPEC.2005.1502528>.

CHEN, C.-M.; CHEN, M.-C.; LI, Y.-L. Mining key formative assessment rules based on learner profiles for web-based learning systems. *Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*, p. 584–588, 2007.

CLEARY, J. G.; TRIGG, L. E. K*: An instance-based learner using and entropic distance measure. In: *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995. (ICML'95), p. 108–114. ISBN 1558603778.

CORAZZA, A.; MARTINO, S. D.; FERRUCCI, F.; GRAVINO, C.; MENDES, E. Investigating the use of support vector regression for web effort estimation. *Empirical Software Engineering*, v. 16, n. 2, p. 211–243, Apr 2011. ISSN 1573-7616.

CRUZ, R. M.; SABOURIN, R.; CAVALCANTI, G. D. Meta-des.oracle: Meta-learning and feature selection for dynamic ensemble selection. *Information Fusion*, v. 38, p. 84 – 103, 2017. ISSN 1566-2535.

CRUZ, R. M.; SABOURIN, R.; CAVALCANTI, G. D. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion*, v. 41, p. 195 – 216, 2018. ISSN 1566-2535.

CRUZ, R. M. O.; SABOURIN, R.; CAVALCANTI, G. D. C. Analyzing dynamic ensemble selection techniques using dissimilarity analysis. In: GAYAR, N. E.; SCHWENKER, F.; SUEN, C. (Ed.). *Artificial Neural Networks in Pattern Recognition*. Cham: Springer International Publishing, 2014. p. 59–70. ISBN 978-3-319-11656-3.

CRUZ, R. M. O.; SABOURIN, R.; CAVALCANTI, G. D. C. Meta-des.h: A dynamic ensemble selection technique using meta-learning and a dynamic weighting approach. In: *2015 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2015. p. 1–8. ISSN 2161-4393.

DELAVARI, N.; PHON-AMNUAISUK, S.; ZADEH, M. B. Data mining application in higher learning institutions. *Informatics in Education*, v. 7, p. 31–54, 04 2008.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, JMLR.org, v. 7, p. 1–30, dez. 2006. ISSN 1532-4435.

DESMARAIS, M.; GAGNON, M. Bayesian student models based on item to item knowledge structures. In: . [S.l.: s.n.], 2006. v. 4227, p. 111–124. ISBN 978-3-540-45777-0.

Di Nucci, D.; Palomba, F.; Oliveto, R.; De Lucia, A. Dynamic selection of classifiers in bug prediction: An adaptive method. *IEEE Transactions on Emerging Topics in Computational Intelligence*, v. 1, n. 3, p. 202–212, 2017.

DIETTERICH, T. Machine learning research: Four current directions. v. 18, 09 1997.

DIETTERICH, T. G. Ensemble methods in machine learning. In: *Proceedings of the First International Workshop on Multiple Classifier Systems*. London, UK, UK: Springer-Verlag, 2000. (MCS '00), p. 1–15.

DRAGICEVIC, S.; CELAR, S.; TURIC, M. Bayesian network model for task effort estimation in agile software development. *Journal of Systems and Software*, v. 127, p. 109 – 119, 2017. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121217300171>.

ETCHELLS, T.; NEBOT, A.; VELLIDO, A.; LISBOA, P. Learning what is important: Feature selection and rule extraction in a virtual course. In: . [S.l.: s.n.], 2006. p. 401–406.

FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. C. P. d. L. F. D. *Inteligência Artificial: Uma Abordagem de Aprendizagem de Máquina*. [S.l.]: LTC, 2011. ISBN 978-85-216-1880-5.

FAUSETT, L.; ELWASIF, W. Predicting performance from test scores using backpropagation and counterpropagation. In: *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*. [S.l.: s.n.], 1994. v. 5, p. 3398–3402 vol.5.

FENG, J. wen. Predicting students' academic performance with decision tree and neural network. In: . [S.l.: s.n.], 2019.

FREUND, Y.; SCHAPIRE, R. E. Experiments with a new boosting algorithm. In: *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1996. (ICML'96), p. 148–156. ISBN 1558604197.

FRIEDMAN, J. H. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, v. 38, n. 4, p. 367–378, 2002. ISSN 0167-9473. Nonlinear Methods and Data Mining. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167947301000652>.

FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 11, n. 1, p. 86–92, 1940. ISSN 00034851.

GARCIA-PEDRAJAS, N.; HERVAS-MARTINEZ, C.; ORTIZ-BOYER, D. Cooperative coevolution of artificial neural network ensembles for pattern classification. *IEEE Transactions on Evolutionary Computation*, v. 9, n. 3, p. 271–302, 2005.

GEDEON, T.; TURNER, H. S. Explaining student grades predicted by a neural network. In: *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*. [S.l.: s.n.], 1993. v. 1, p. 609–612 vol.1.

GENCEL, C.; BUGLIONE, L. Do base functional component types affect the relationship between software functional size and effort? In: CUADRADO-GALLEGO, J. J.; BRAUNGARTEN, R.; DUMKE, R. R.; ABRAN, A. (Ed.). *Software Process and Product Measurement*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 72–85. ISBN 978-3-540-85553-8.

GRAY, A. R.; MACDONELL, S. G. A comparison of techniques for developing predictive models of software metrics. *Information and Software Technology*, v. 39, n. 6, p. 425 – 437, 1997. ISSN 0950-5849. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0950584996000067>.

GRIMSHAW, S. D. An introduction to the bootstrap. *Technometrics*, Taylor and Francis, v. 37, n. 3, p. 340–341, 1995. Disponível em: <https://doi.org/10.1080/00401706.1995.10484340>.

GUEVARA, F. G.-L. de; FERNáNDEZ-DIEGO, M.; LOKAN, C. The usage of isbsg data fields in software effort estimation: A systematic mapping study. *Journal of Systems and Software*, v. 113, p. 188 – 215, 2016. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121215002642>.

HäMäLäINEN, W.; VINNI, M. Comparison of machine learning methods for intelligent tutoring systems. In: *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*. Berlin, Heidelberg: Springer-Verlag, 2006. (ITS'06), p. 525–534. ISBN 3540351590. Disponível em: <https://doi.org/10.1007/11774303_52>.

HANSEN, L.; SALAMON, P. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 12, n. 10, p. 993–1001, 1990.

HARRINGTON, P. *Machine Learning in Action*. [S.l.]: Manning, 2012.

HELLAS, A.; IHANTOLA, P.; PETERSEN, A.; AJANOVSKI, V. V.; GUTICA, M.; HYNNINEN, T.; KNUTAS, A.; LEINONEN, J.; MESSOM, C.; LIAO, S. N. Predicting academic performance: A systematic literature review. In: *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*. New York, NY, USA: Association for Computing Machinery, 2018. (ITiCSE 2018 Companion), p. 175–199. ISBN 9781450362238. Disponível em: <https://doi.org/10.1145/3293881.3295783>.

HIEN, N.; HADDAWY, P. A decision support system for evaluating international student applications. In: . [S.l.: s.n.], 2007. p. F2A–1.

HO, T. K. Random decision forests. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. [S.l.: s.n.], 1995. v. 1, p. 278–282 vol.1.

HO, T. K. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 20, n. 8, p. 832–844, Aug 1998. ISSN 0162-8828.

HO, T. K. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 20, n. 8, p. 832–844, 1998.

HOLLANDER, M.; WOLFE, D. A.; CHICKEN, E. *Nonparametric Statistical Methods*. 3. ed. [S.l.]: Wiley, 2013. Hardcover. ISBN 0470387378.

HOLTE, R. C. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, Kluwer, v. 11, p. 63–91, 1993.

HOSNI, M.; A.IDRI; NASSIF, A. B.; ABRAN, A. Heterogeneous ensembles for software development effort estimation. In: *2016 3rd International Conference on Soft Computing Machine Intelligence (ISCMI)*. [S.l.: s.n.], 2016. p. 174–178.

HOSNI, M.; IDRI, A.; ABRAN, A. Investigating heterogeneous ensembles with filter feature selection for software effort estimation. In: *Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement*. New York, NY, USA: Association for Computing Machinery, 2017. (IWSM Mensura '17), p. 207–220. ISBN 9781450348539. Disponível em: <https://doi.org/10.1145/3143434.3143456>.

HOSNI, M.; IDRI, A.; ABRAN, A. Improved effort estimation of heterogeneous ensembles using filter feature selection. In: . [S.l.: s.n.], 2018. p. 405–412.

HOSNI, M.; IDRI, A.; ABRAN, A.; NASSIF, A. On the value of parameter tuning in heterogeneous ensembles effort estimation. *Soft Computing*, p. 1–34, 2017. Cited By 0; Article in Press. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85035807309&doi=10.1007\%2fs00500-017-2945-4&partnerID=40&md5=83d094f56b80a21295cf4230621f445d>.

HSU, J. C. *Multiple Comparisons: Theory and Methods*. [S.l.]: Chapman and Hall/CRC, 1996. ISBN 0-412-98281-1.

IDRI, A.; HOSNI, M.; ABRAN, A. Systematic literature review of ensemble effort estimation. *Journal of Systems and Software*, v. 118, p. 151 – 175, 2016. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121216300450>.

IDRI, A.; HOSNI, M.; ABRAN, A. Systematic mapping study of ensemble effort estimation. In: *ENASE*. [S.l.: s.n.], 2016.

INUSAH, F. Data mining and visualisation of basic educational resources for quality education. *International Journal of Engineering Trends and Technology*, v. 70, p. 296–307, 12 2022.

JOHN, G. H.; LANGLEY, P. Estimating continuous distributions in bayesian classifiers. *CoRR*, abs/1302.4964, 2013. Disponível em: <http://arxiv.org/abs/1302.4964>.

JORGENSEN, M.; SHEPPERD, M. A systematic review of software development cost estimation studies. *IEEE Transactions on Software Engineering*, v. 33, n. 1, p. 33–53, Jan 2007. ISSN 0098-5589.

KESELMAN, H.; KESELMAN, J.; GAMES, P. A. Maximum familywise type i error rate: The least significant difference. *Newman-Keuls*, 1991.

KITTLER, J.; HATEF, M.; DUIN, R. P. W.; MATAS, J. On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.*, IEEE Computer Society, Washington, DC, USA, v. 20, n. 3, p. 226–239, mar. 1998. ISSN 0162-8828.

KO, A. H. R.; SABOURIN, R.; BRITTO JR., A. S. From dynamic classifier selection to dynamic ensemble selection. *Pattern Recogn.*, Elsevier Science Inc., New York, NY, USA, v. 41, n. 5, p. 1718–1731, maio 2008. ISSN 0031-3203.

KO, A. H. R.; SABOURIN, R.; JR., A. B. K-nearest oracle for dynamic ensemble selection. In: *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*. [S.l.: s.n.], 2007. v. 1, p. 422–426. ISSN 1520-5363.

KOCAGUNELI, E.; MENZIES, T.; KEUNG, J. W. On the value of ensemble effort estimation. *IEEE Transactions on Software Engineering*, v. 38, n. 6, p. 1403–1416, Nov 2012. ISSN 0098-5589.

KOHAVI, R. The power of decision tables. In: *Proceedings of the 8th European Conference on Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 1995. (ECML '95), p. 174–189. ISBN 3540592865.

KOTSIANTIS, S. Use of machine learning techniques for educational proposes: a decision support system for forecasting students' grades. *Artificial Intelligence Review*, n. 37, p. 331–344, 2012.

KOTSIANTIS, S.; PATRIARCHEAS, K.; XENOS, M. A combinational incremental ensemble of classifiers as a technique for predicting students' performance in distance education. *Knowledge-Based Systems*, v. 23, n. 6, p. 529–535, 2010. ISSN 0950-7051. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0950705110000456>.

KOTSIANTIS, S.; PINTELAS, P. Predicting students marks in hellenic open university. In: *Fifth IEEE International Conference on Advanced Learning Technologies (ICALT'05)*. [S.l.: s.n.], 2005. p. 664–668.

KRUSKAL, W. H.; WALLIS, W. A. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, Taylor & Francis, v. 47, n. 260, p. 583–621, 1952. Disponível em: <https://amstat.tandfonline.com/doi/abs/10.1080/01621459.1952.10483441>.

KULTUR, Y.; TURHAN, B.; BENER, A. Ensemble of neural networks with associative memory (enna) for estimating software development costs. *Know.-Based Syst.*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 22, n. 6, p. 395–402, ago. 2009. ISSN 0950-7051.

KUNCHEVA, L. A theoretical study on six classifier fusion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 2, p. 281–286, 2002.

LANDWEHR, N.; HALL, M.; FRANK, E. Logistic model trees. *Machine Learning*, v. 59, p. 161–205, 02 2005.

LIVIERIS, I.; DRAKOPOULOU, K.; PINTELAS, P. Predicting students' performance using artificial neural networks'. In: . [S.l.: s.n.], 2012.

LYNN, N. D.; EMANUEL, A. W. R. Using data mining techniques to predict students' performance. a review. *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, v. 1096, n. 1, p. 012083, mar 2021. Disponível em: <https://dx.doi.org/10.1088/1757-899X/1096/1/012083>.

LóPEZ-MARTíN, C.; ABRAN, A. Neural networks for predicting the duration of new software projects. *Journal of Systems and Software*, v. 101, p. 127 – 135, 2015. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121214002805>.

MACKAY, D. J. *Introduction to Gaussian Processes*. Dept. of Physics, 1998. Disponível em: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.81.1927&rep=rep1&type=pdf>.

MALGONDE, O.; CHARI, K. An ensemble-based model for predicting agile software development effort. *Empirical Software Engineering*, v. 24, 04 2019.

MARTÍNEZ, D. C. Predicting student outcomes using discriminant function analysis. In: . [S.l.: s.n.], 2001.

MATOS DANIEL ABUD SEABRA; RODRIGUES, E. C. *Análise fatorial*. [S.l.]: Enap, 2019. ISBN 978-85-256-0118-6.

MENDES, E.; WATSON, I.; TRIGGS, C.; MOSLEY, N.; COUNSELL, S. A comparative study of cost estimation models for web hypermedia applications. *Empirical Software Engineering*, v. 8, n. 2, p. 163–196, Jun 2003. ISSN 1573-7616. Disponível em: <https://doi.org/10.1023/A:1023062629183>.

MINAEI-BIDGOLI, B.; KASHY, D.; KORTEMEYER, G.; PUNCH, W. Predicting student performance: an application of data mining methods with an educational web-based system. In: *33rd Annual Frontiers in Education, 2003. FIE 2003.* [S.l.: s.n.], 2003. v. 1, p. T2A–13.

MINKU, L. L.; YAO, X. Ensembles and locality: Insight on improving software effort estimation. *Information and Software Technology*, v. 55, n. 8, p. 1512 – 1528, 2013. ISSN 0950-5849.

MITCHELL, T. M. *Machine Learning*. [S.l.]: McGraw-Hill, 1997. v. 1.

MONARD, M. C.; BARANAUSKAS, J. A. *Conceitos sobre Aprendizado de Máquina*. [S.l.]: Manole, 2003. 89-114 p.

MOREIRA, J. M.; JORGE, A. M.; SOARES, C.; SOUSA, J. F. de. Ensemble learning: A study on different variants of the dynamic selection approach. In: PERNER, P. (Ed.). *Machine Learning and Data Mining in Pattern Recognition*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 191–205. ISBN 978-3-642-03070-3.

MOREIRA, J. M.; SOARES, C.; JORGE, A.; SOUSA, J. F. D. Ensemble approaches for regression: A survey. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 45, n. 1, p. 10:1–10:40, dez. 2012. ISSN 0360-0300.

MORETTIN, P. A.; BUSSAB, W. de O. *Estatística Básica*. 6. ed. São Paulo, SP, Brazil: Saraiva, 2010. ISBN 978-85-02-08177-2.

MOURA, T. J.; CAVALCANTI, G. D.; OLIVEIRA, L. S. Mine: A framework for dynamic regressor selection. *Information Sciences*, v. 543, p. 157–179, 2021. ISSN 0020-0255. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020025520307192>.

MOURA, T. J. M.; CAVALCANTI, G. D. C.; OLIVEIRA, L. S. Evaluating competence measures for dynamic regressor selection. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2019. p. 1–8.

MYLLER, N.; SUHONEN, J.; SUTINEN, E. Using data mining for improving web-based course design. In: *International Conference on Computers in Education, 2002. Proceedings.* [S.l.: s.n.], 2002. p. 959–963 vol.2.

NAFIE, F.; HAMED, A. Using data mining techniques in building a model to determine the factors affecting academic data for undergraduate students. p. 306, 04 2021.

NASCIMENTO, R. L. S. do; FAGUNDES, R. A. A.; MACIEL, A. M. A. Prediction of school efficiency rates through ensemble regression application. In: *2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT)*. [S.l.: s.n.], 2019. v. 2161-377X, p. 194–198.

NEBOT, A.; ESPINOZA, F. C. An e-learning toolbox based on rule-based fuzzy approaches. *Applied Sciences*, v. 10, p. 6804, 09 2020.

NEBOT, A.; ESPINOZA, F. C.; VELLIDO, A. Identification of fuzzy models to predict students performance in an e-learning environment. *Proceedings of the Fifth IASTED International Conference on Web-based Education*, v. 2006, p. 74–79, 01 2006.

OGOR, E. N. Student academic performance monitoring and evaluation using data mining techniques. In: *Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007)*. [S.l.: s.n.], 2007. p. 354–359.

OLIVEIRA, A. L. Estimation of software project effort with support vector regression. *Neurocomputing*, v. 69, n. 13, p. 1749 – 1753, 2006. ISSN 0925-2312.

OLIVEIRA, A. L.; BRAGA, P. L.; LIMA, R. M.; CORNéLIO, M. L. Ga-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation. *Information and Software Technology*, v. 52, n. 11, p. 1155 – 1166, 2010. ISSN 0950-5849.

PALANISWAMY, S.; VENKATESAN, R. Hyperparameters tuning of ensemble model for software effort estimation. *Journal of Ambient Intelligence and Humanized Computing*, p. 1–11, 2020.

PARDOS, Z.; HEFFERNAN, N.; RUIZ, C.; BECK, J. The composition effect: Conjuntive or compensatory? an analysis of multi-skill math questions in its. In: . [S.l.: s.n.], 2008. p. 147–156.

PARDOS, Z. A.; HEFFERNAN, N. T.; ANDERSON, B.; HEFFERNAN, C. L. The effect of model granularity on student performance prediction using bayesian networks. In: CONATI, C.; MCCOY, K.; PALIOURAS, G. (Ed.). *User Modeling 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 435–439. ISBN 978-3-540-73078-1.

PARTALAS, I.; TSOUMAKAS, G.; HATZIKOS, E. V.; VLAHAVAS, I. Greedy regression ensemble selection: Theory and an application to water quality prediction. *Information Sciences*, v. 178, n. 20, p. 3867–3879, 2008. ISSN 0020-0255. Special Issue on Industrial Applications of Neural Networks. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020025508001576>.

PERRONE, M.; COOPER, L. When networks disagree: Ensemble methods for hybrid neural networks. *Neural networks for speech and image processing*, 08 1993.

PHANNACHITTA, P.; MATSUMOTO, K. Model-based software effort estimation–a robust comparison of 14 algorithms widely used in the data science community. *International Journal of Innovative Computing, Information and Control*, v. 15, n. 2, 04 2019.

PLATT, J. Fast training of support vector machines using sequential minimal optimization. In: *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998. Disponível em: <https://www.microsoft.com/en-us/research/publication/fast-training-of-support-vector-machines-using-sequential-minimal-optimization/>.

POSPIESZNY, P.; CZARNACKA-CHROBOT, B.; KOBYLINSKI, A. An effective approach for software project effort and duration estimation with machine learning algorithms. *Journal of Systems and Software*, v. 137, p. 184 – 196, 2018. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121217302947>.

PRITCHARD, D.; WARNAKULASOORIYA, R. Data from a web-based homework tutor can predict student's final exam score. 01 2005.

QUINLAN, J. R. Learning with continuous classes. In: *Proceedings of Australian Joint Conference on Artificial Intelligence*. [S.l.]: World Scientific, 1992. p. 343–348.

QUINLAN, J. R. *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. ISBN 1558602380.

RANGONE, G.; PIZARRO, C.; MONTEJANO, G. Automation of an educational data mining model applying interpretable machine learning and auto machine learning. In: _____. [S.l.: s.n.], 2022. p. 22–30. ISBN 978-981-16-5791-7.

RAO, K.; RAO, G. Ensemble learning with recursive feature elimination integrated software effort estimation: a novel approach. *Evolutionary Intelligence*, v. 14, 03 2020.

ROMERO, C.; VENTURA, S. Educational data mining: A review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, v. 40, n. 6, p. 601–618, 2010.

ROMERO, C.; VENTURA, S. Educational data mining: A review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, v. 40, n. 6, p. 601–618, 2010.

ROMERO, C.; VENTURA, S. Data mining in education. *Wiley Int. Rev. Data Min. and Knowl. Disc.*, John Wiley and amp; Sons, Inc., USA, v. 3, n. 1, p. 12–27, jan 2013. ISSN 1942-4787. Disponível em: <https://doi.org/10.1002/widm.1075>.

ROMERO, C.; VENTURA, S.; ESPEJO, P. G.; HERVÁS-MARTÍNEZ, C. Data mining algorithms to classify students. In: *Educational Data Mining*. [S.l.: s.n.], 2008.

ROONEY, N.; PATTERSON, D. A weighted combination of stacking and dynamic integration. *Pattern Recognition*, v. 40, n. 4, p. 1385–1388, 2007. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320306004262>.

ROONEY, N.; PATTERSON, D.; ANAND, S.; TSYMBAL, A. Dynamic integration of regression models. In: ROLI, F.; KITTLER, J.; WINDEATT, T. (Ed.). *Multiple Classifier Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 164–173.

ROUSSEEUW, P.; LEROY, A. Robust regression & outlier detection, john wiley & sons. *Journal of Educational Statistics*, v. 13, p. 358–364, 01 1988.

ROUSSEEUW, P. J.; LEROY, A. M. *Robust Regression and Outlier Detection*. 1. ed. Antwerp, Belgium: John Wiley & Sons, 1987. ISBN 978-04-71-85233-9.

RUSSELL, S. J.; NORVIG, P. *Artificial intelligence: a modern approach*. 3. ed. [S.l.]: Prentice-Hall, Inc., 2010. ISBN 85-346-0122-4.

SANTOS, A.; RODRíGUEZ, A. I.; PINTO-LLORENTE, A. Identification of characteristics and functionalities for the design of an academic analytics model for higher education. In: . [S.l.: s.n.], 2020. p. 997–1003.

SCACCIA, K. *Validação Cruzada Aninhada com Scikit-learn*. 2020. <https://dataml.com.br/validacao-cruzada-aninhada-com-scikit-learn/>. [Online; acessado em 15-06-2021].

SCHAPIRE, R. E.; FREUND, Y.; BARTLETT, P.; LEE, W. S. Boosting the margin: a new explanation for the effectiveness of voting methods. *Ann. Statist.*, The Institute of Mathematical Statistics, v. 26, n. 5, p. 1651–1686, 10 1998.

SENI, G.; ELDER, J. F. Ensemble methods in data mining: Improving accuracy through combining predictions. *Morgan and Claypool Publishers*, 2010. Disponível em: <https://www.morganclaypool.com/doi/abs/10.2200/S00240ED1V01Y200912DMK002>.

SERGIO, A. T.; LIMA, T. P. D.; LUDERMIR, T. B. Dynamic selection of forecast combiners. *Neurocomputing*, v. 218, p. 37–50, 2016. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231216309687>.

SHAH, M. A.; JAWAWI, D. N. A.; ISA, M. A.; YOUNAS, M.; ABDELMABOUD, A.; SHOLICHIN, F. Ensembling artificial bee colony with analogy-based estimation to improve software development effort prediction. *IEEE Access*, v. 8, p. 58402–58415, 2020.

SHAHIRI, A. M.; HUSAIN, W.; RASHID, N. A. A review on predicting student's performance using data mining techniques. *Procedia Computer Science*, v. 72, p. 414–422, 2015. ISSN 1877-0509. The Third Information Systems International Conference 2015. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1877050915036182>.

SHANGPING, D.; PING, Z. A data mining algorithm in distance learning. In: *2008 12th International Conference on Computer Supported Cooperative Work in Design*. [S.l.: s.n.], 2008. p. 1014–1017.

SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples). *Biometrika*, [Oxford University Press, Biometrika Trust], v. 52, n. 3/4, p. 591–611, 1965. ISSN 00063444. Disponível em: <http://www.jstor.org/stable/2333709>.

SHEPPERD, M.; KADODA, G. Comparing software prediction techniques using simulation. *IEEE Transactions on Software Engineering*, v. 27, n. 11, p. 1014–1022, Nov 2001. ISSN 0098-5589.

SHEPPERD, M.; MACDONELL, S. Evaluating prediction systems in software project estimation. *Information and Software Technology*, v. 54, n. 8, p. 820 – 827, 2012. ISSN 0950-5849. Special Issue: Voice of the Editorial Board. Disponível em: <http://www.sciencedirect.com/science/article/pii/S095058491200002X>.

SHEPPERD, M.; SCHOFIELD, C. Estimating software project effort using analogies. *IEEE Transactions on Software Engineering*, v. 23, n. 11, p. 736–743, Nov 1997. ISSN 0098-5589.

SHEVADE, S.; KEERTHI, S.; BHATTACHARYYA, C.; MURTHY, K. Improvements to the smo algorithm for svm regression. *IEEE Transactions on Neural Networks*, v. 11, n. 5, p. 1188–1193, 2000.

SHI, H. *Best-first Decision Tree Learning*. Dissertação (Mestrado) — The University of Waikato, New Zealand, 2007. Disponível em: <https://hdl.handle.net/10289/2317>.

SHIRABAD, J. S.; MENZIES, T. J. *The PROMISE Repository of Software Engineering Databases.* 2005. School of Information Technology and Engineering, University of Ottawa, Canada.

SHUNMUGAPRIYA, P.; KANMANI, S. Optimization of stacking ensemble configurations through artificial bee colony algorithm. *Swarm and Evolutionary Computation*, v. 12, n. Supplement C, p. 24 – 32, 2013. ISSN 2210-6502.

SINGHAL, G. *Ensemble Methods in Machine Learning: Bagging Versus Boosting*. 2020. <https://www.pluralsight.com/guides/ensemble-methods:-bagging-versus-boosting>. [Online; acessado em 15-06-2021].

SPIKOL, D.; RUFFALDI, E.; CUKUROVA, M. Using multimodal learning analytics to identify aspects of collaboration in project-based learning. In: . [S.l.: s.n.], 2017.

STENSRUD, E. Alternative approaches to effort prediction of erp projects. *Information and Software Technology*, v. 43, n. 7, p. 413 – 423, 2001. ISSN 0950-5849. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0950584901001471>.

STEVENS, R.; SOLLER, A.; GIORDANI, A.; GEROSA, L.; COOPER, M.; COX, C. Developing a framework for integrating prior problem solving and knowledge sharing histories of a group to predict future group performance. In: *2005 International Conference on Collaborative Computing: Networking, Applications and Worksharing*. [S.l.: s.n.], 2005. p. 9 pp.–.

SUITS, D. B. Use of dummy variables in regression equations. *Journal of the American Statistical Association*, Taylor & Francis, v. 52, n. 280, p. 548–551, 1957. Disponível em: <https://www.tandfonline.com/doi/abs/10.1080/01621459.1957.10501412>.

THOMAS, E.; GALAMBOS, N. What satisfies students? mining student-opinion data with regression and decision tree analysis. *Research in Higher Education*, v. 45, p. 251–269, 05 2004.

TORRES-SOSPEDRA, J.; HERNÁNDEZ-ESPINOSA, C.; FERNÁNDEZ-REDONDO, M. Adaptive boosting: Dividing the learning set to increase the diversity and performance of the ensemble. In: KING, I.; WANG, J.; CHAN, L.-W.; WANG, D. (Ed.). *Neural Information Processing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 688–697.

TRENDOWICZ, R. J. A. Wideband delphi. in: Software project effort estimation. In: _____. [S.l.]: Springer, Cham., 2014.

TWALA, B.; VERNER, J. Toward accurate software effort prediction using multiple classifier systems. *Computational Intelligence and Quantitative Software Engineering*, p. 135–151, 2016.

UKWUOMA, C.; BO, C.; CHIKWENDU, I.; BONDZIE-SELBY, E. Performance analysis of students based on data mining techniques: A literature review. In: . [S.l.: s.n.], 2019. p. 1–5.

USMAN, M.; MENDES, E.; WEIDT, F.; BRITTO, R. Effort estimation in agile software development: A systematic literature review. In: *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*. New York, NY, USA: ACM, 2014. (PROMISE '14), p. 82–91. ISBN 978-1-4503-2898-2.

WAIKATO, U. *Class ConjunctiveRule*. 2022. Disponível em: <https://weka.sourceforge.io/doc.packages/conjunctiveRule/weka/classifiers/rules/ConjunctiveRule.html>.

WAIKATO, U. *Class DecisionStump*. 2022. Disponível em: <https://weka.sourceforge.io/doc.dev/weka/classifiers/functions/LinearRegression.html>.

WAIKATO, U. *Class Logistic Regression*. 2022. Disponível em: <https://weka.sourceforge.io/doc.dev/weka/classifiers/functions/Logistic.html>.

WAIKATO, U. *Class Multilayer Perceptron*. 2022. Disponível em: <https://weka.sourceforge.io/doc.dev/weka/classifiers/functions/MultilayerPerceptron.html>.

WAIKATO, U. *Class RBFNetWork*. 2022. Disponível em: <https://weka.sourceforge.io/doc.stable/weka/classifiers/functions/RBFNetwork.html>.

WAIKATO, U. *Class RBFNetWork*. 2022. Disponível em: <https://weka.sourceforge.io/doc.dev/weka/classifiers/trees/REPTree.html>.

WAIKATO, U. *Class RBFNetWork*. 2022. Disponível em: <https://weka.sourceforge.io/doc.stable-3-8/weka/classifiers/rules/ZeroR.html>.

WAIKATO, U. *Weka Documentation*. 2022. Disponível em: <https://docs.weka.io/>.

WALKERDEN, F.; JEFFERY, R. An empirical study of analogy-based software effort estimation. *Empirical Software Engineering*, v. 4, n. 2, p. 135–158, Jun 1999. ISSN 1573-7616. Disponível em: <https://doi.org/10.1023/A:1009872202035>.

WANG, A.; NEWLIN, M. Predictors of web-student performance: The role of self-efficacy and reasons for taking an on-line class. *Computers in Human Behavior*, v. 18, p. 151–163, 03 2002.

WANG, T.; MITROVIC, A. Using neural networks to predict student's performance. In: *International Conference on Computers in Education, 2002. Proceedings.* [S.l.: s.n.], 2002. p. 969–973 vol.2.

WANG, Y.; WITTEN, I. H. *Induction of model trees for predicting continuous classes.* Hamilton, New Zealand: University of Waikato, Department of Computer Science, 1996. Disponível em: <https://researchcommons.waikato.ac.nz/handle/10289/1183>.

WANG, Y.-h.; LIAO, H.-C. Data mining for adaptive learning in a tesl-based e-learning system. *Expert Syst. Appl.*, v. 38, p. 6480–6485, 06 2011.

WEBB, G. I.; ZHENG, Z. Multistrategy ensemble learning: reducing error by combining ensemble learning techniques. *IEEE Transactions on Knowledge and Data Engineering*, v. 16, n. 8, p. 980–991, Aug 2004. ISSN 1041-4347.

WEN, J.; LI, S.; LIN, Z.; HU, Y.; HUANG, C. Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, v. 54, n. 1, p. 41 – 59, 2012. ISSN 0950-5849.

WITTEN, I. H.; FRANK, E. *Data Mining: Practical Machine Learning Tools and Techniques*. 1. ed. [S.l.]: Morgan Kaufmann Publishers Inc., 2011.

WOLPERT, D.; MACREADY, W. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, v. 1, n. 1, p. 67–82, 1997.

WOLPERT, D. H. Stacked generalization. *Neural Networks*, v. 5, n. 2, p. 241 – 259, 1992. ISSN 0893-6080. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0893608005800231>.

WOODS, K.; KEGELMEYER, W. P.; BOWYER, K. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 19, n. 4, p. 405–410, Apr 1997. ISSN 0162-8828.

YAVUZ, E. An evaluation of web based instruction in view of the tutors' and students' perspectives. *The Turkish Online Journal of Distance Education*, v. 9, 04 2008.

APÊNDICE A – DISTRIBUIÇÃO DOS DADOS

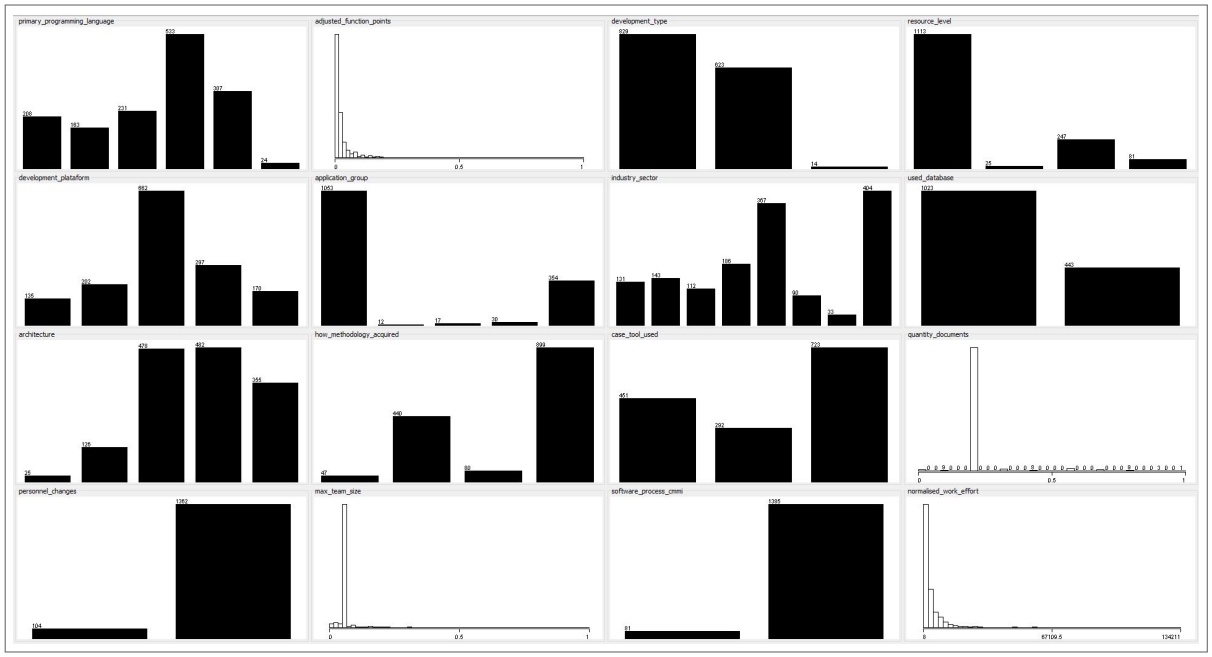


Figura 48 – Distribuição dos dados do conjunto ISBSG.

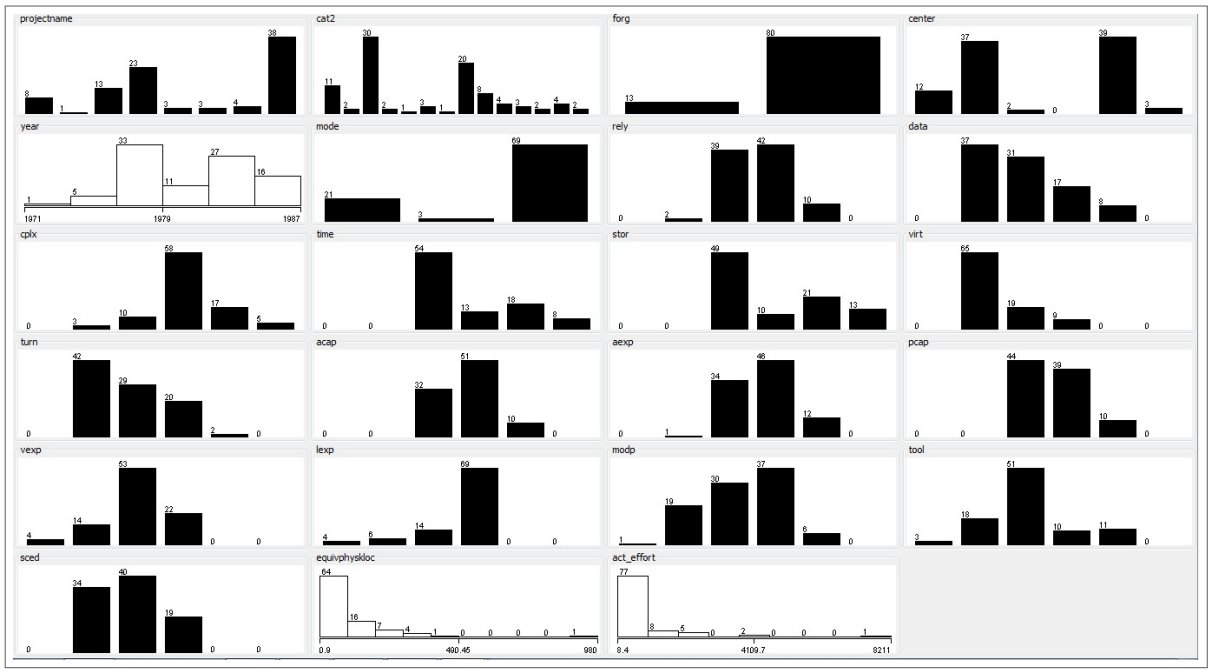


Figura 49 – Distribuição dos dados do conjunto Cocomonasa V2.

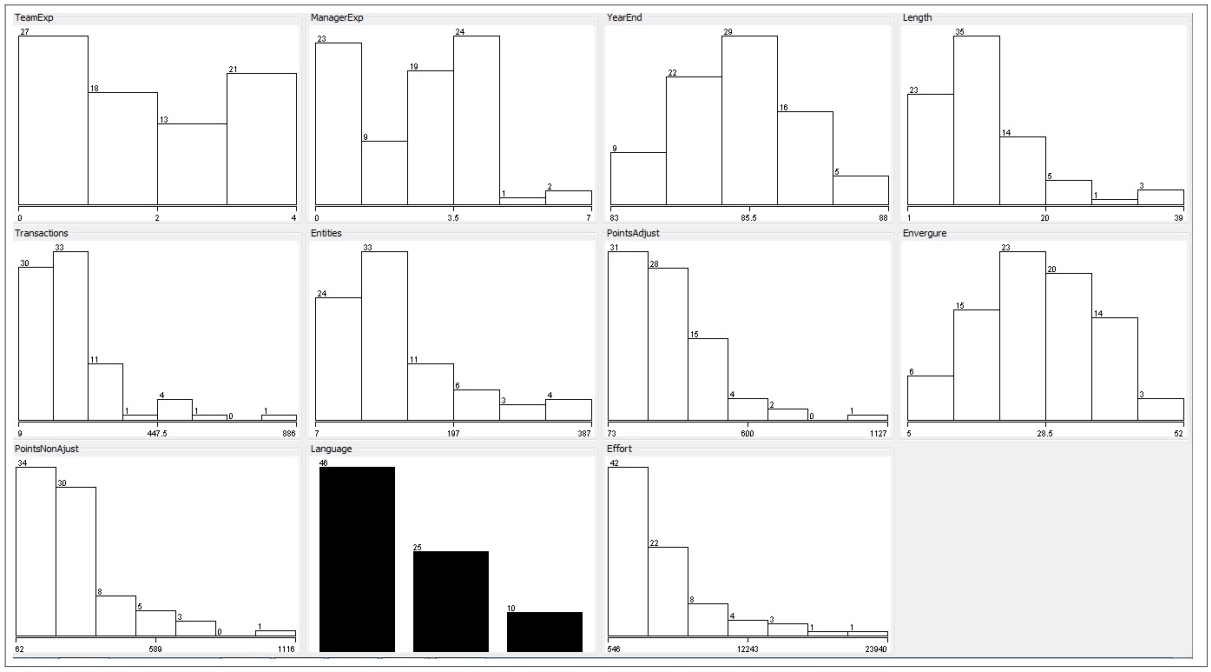


Figura 50 – Distribuição dos dados do conjunto Desharnais.

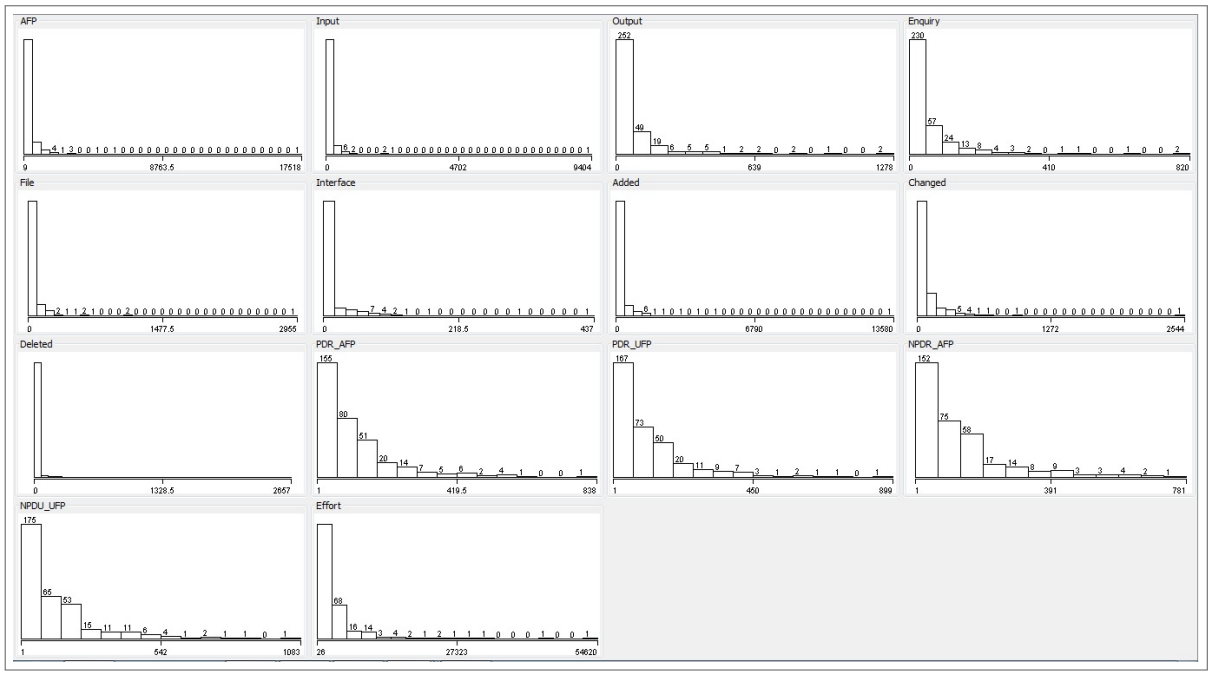


Figura 51 – Distribuição dos dados do conjunto China.

Figura 52 – Distribuição dos dados do conjunto Cocomonasa V1.

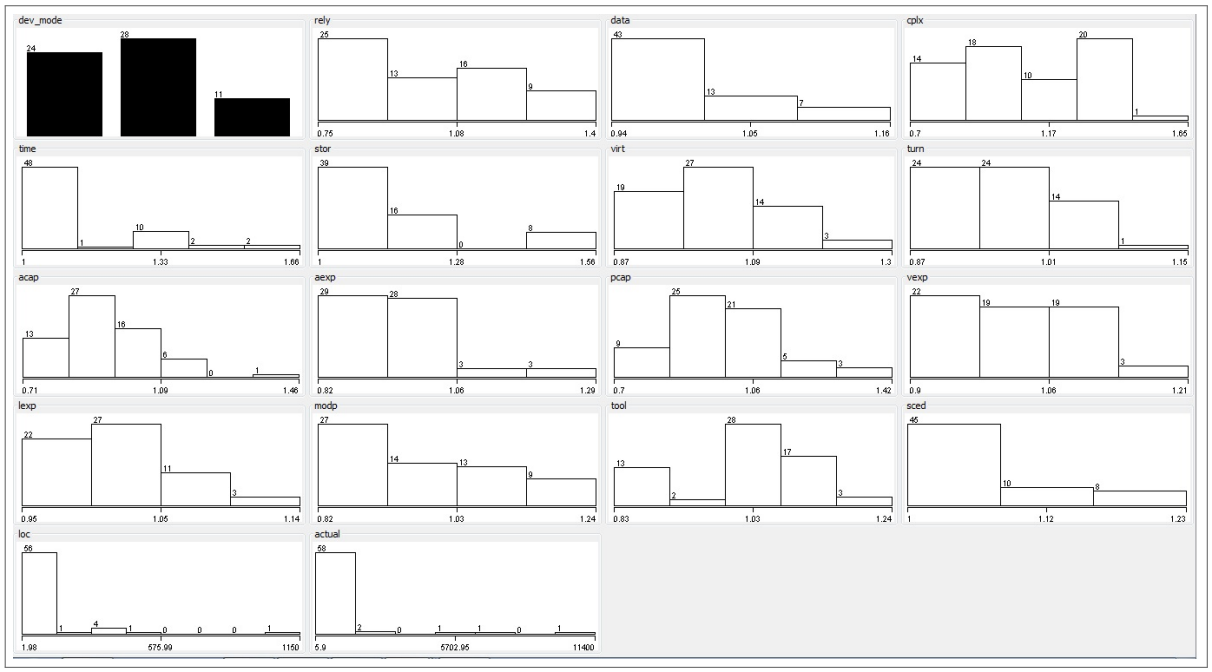


Figura 53 – Distribuição dos dados do conjunto Cocomo81.

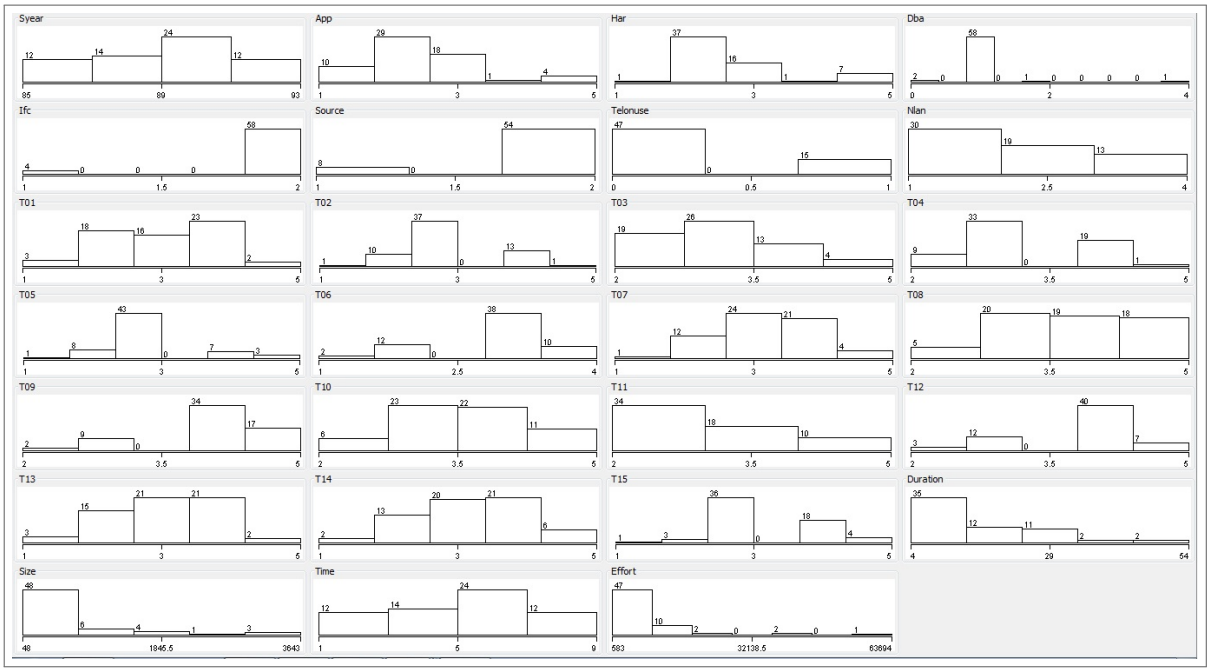


Figura 54 – Distribuição dos dados do conjunto Maxwell.

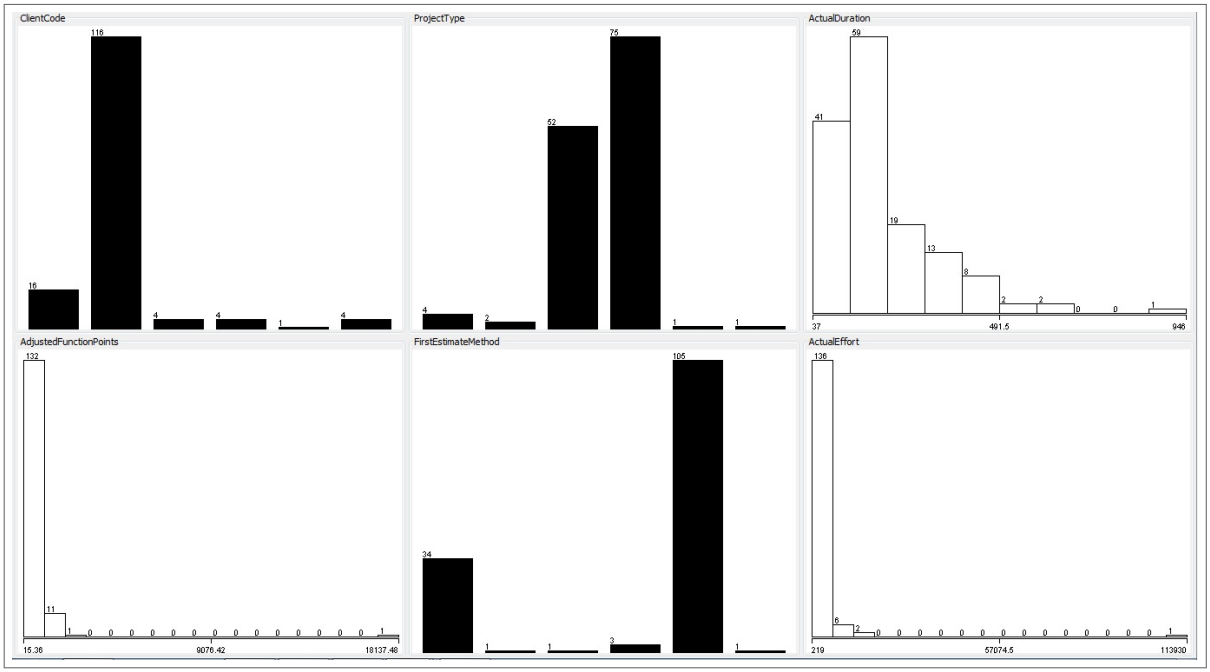


Figura 55 – Distribuição dos dados do conjunto Kitchenham.

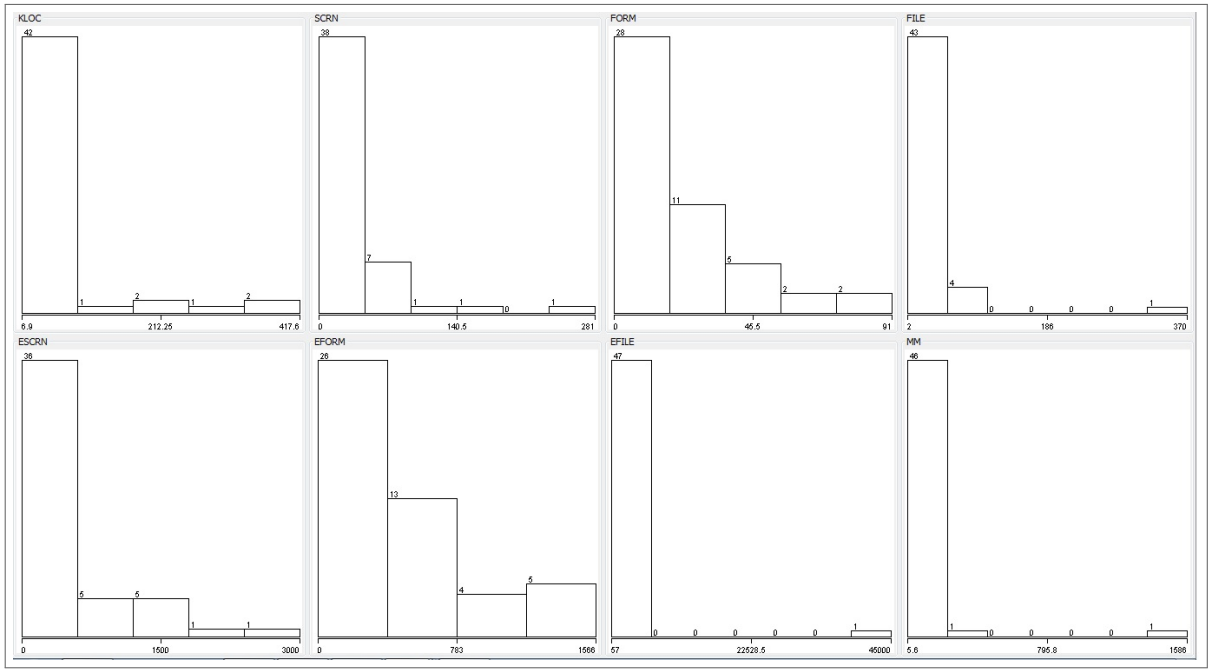


Figura 56 – Distribuição dos dados do conjunto Miyazaki94.

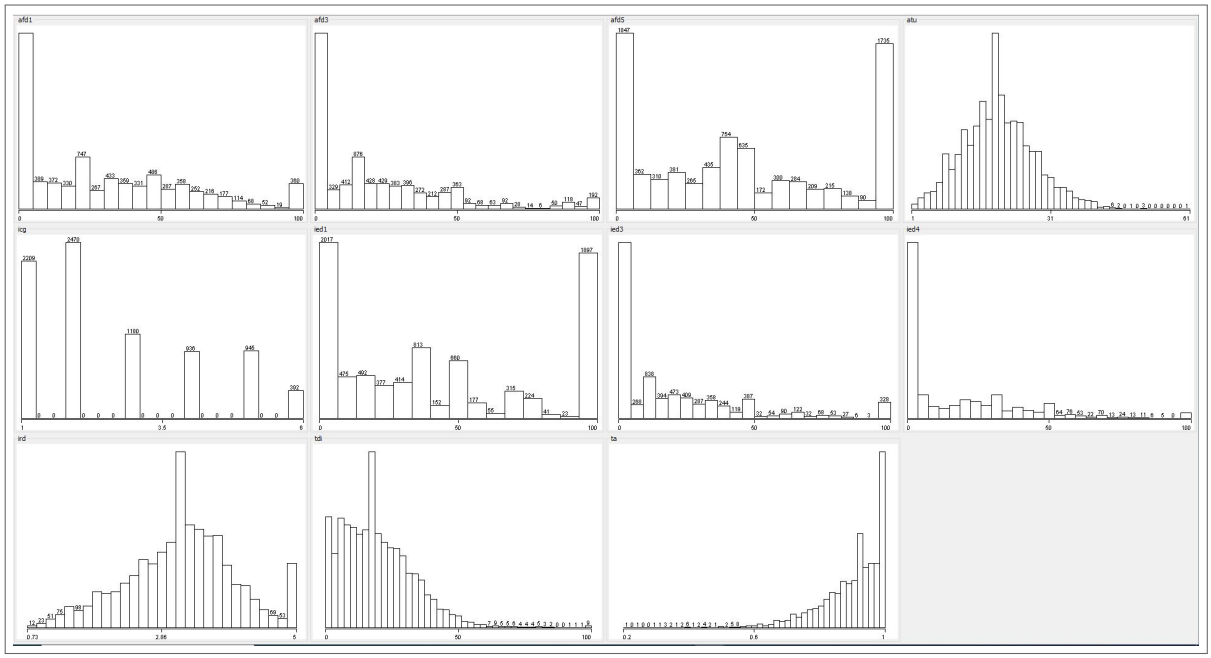


Figura 57 – Distribuição dos dados do conjunto de taxas de aprovação do ensino fundamental.

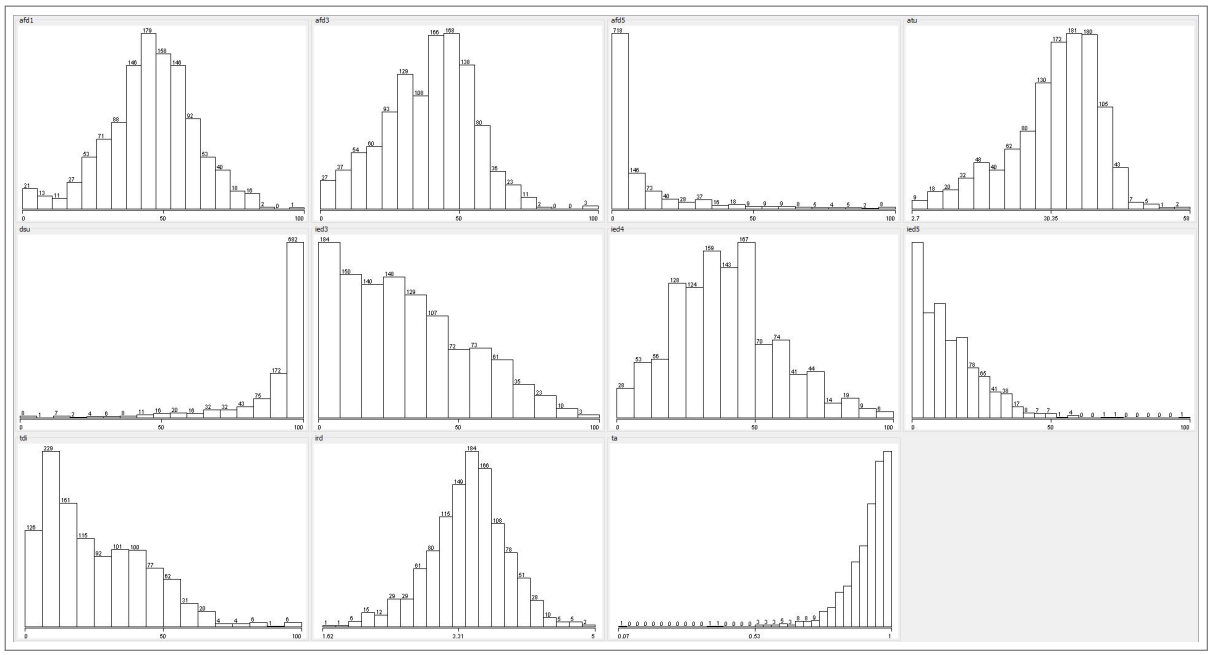


Figura 58 – Distribuição dos dados do conjunto de taxas de aprovação do ensino médio.

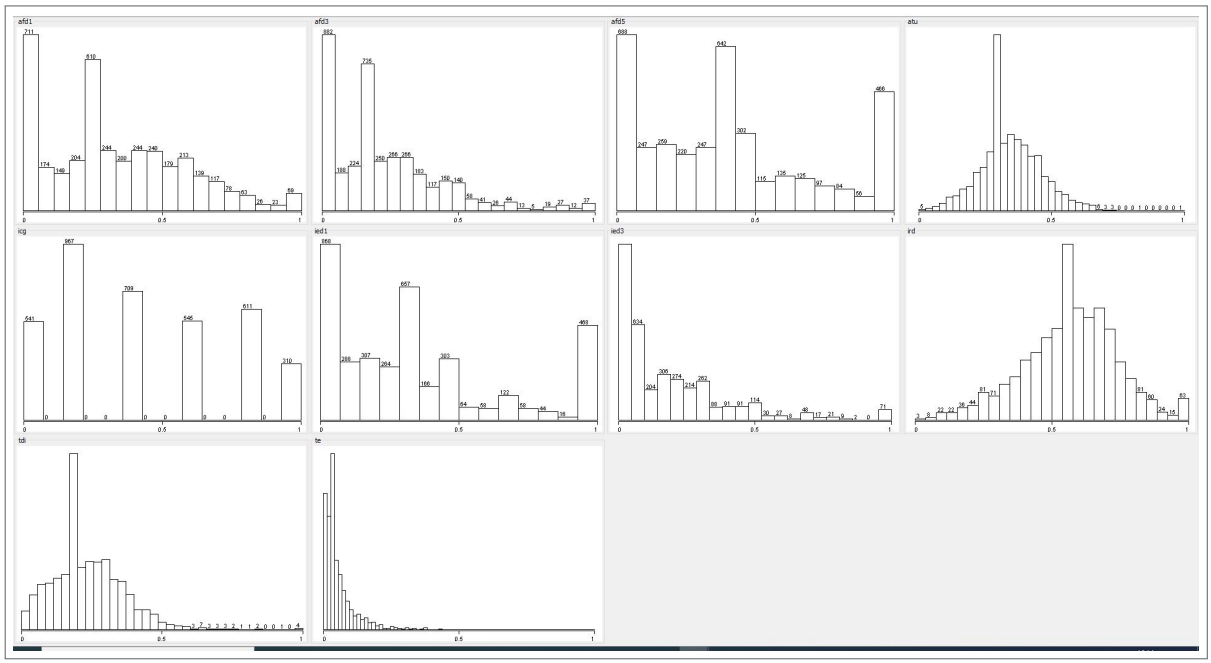


Figura 59 – Distribuição dos dados do conjunto de taxas de evasão do ensino fundamental.

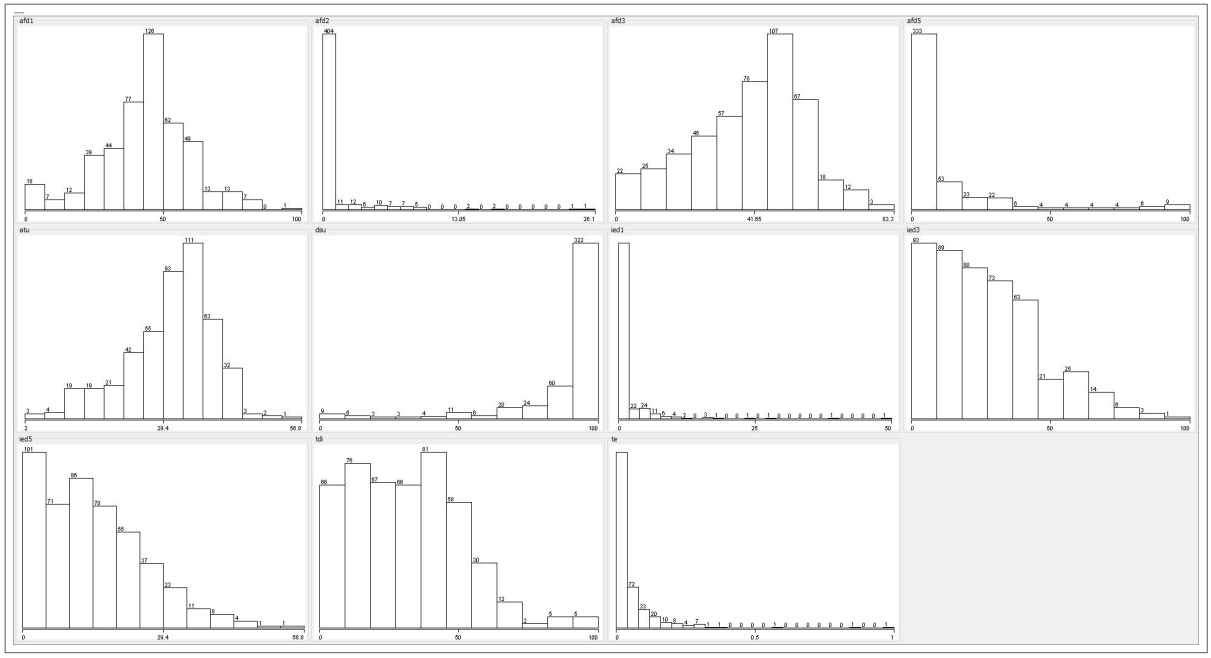


Figura 60 – Distribuição dos dados do conjunto de taxas de evasão do ensino médio.

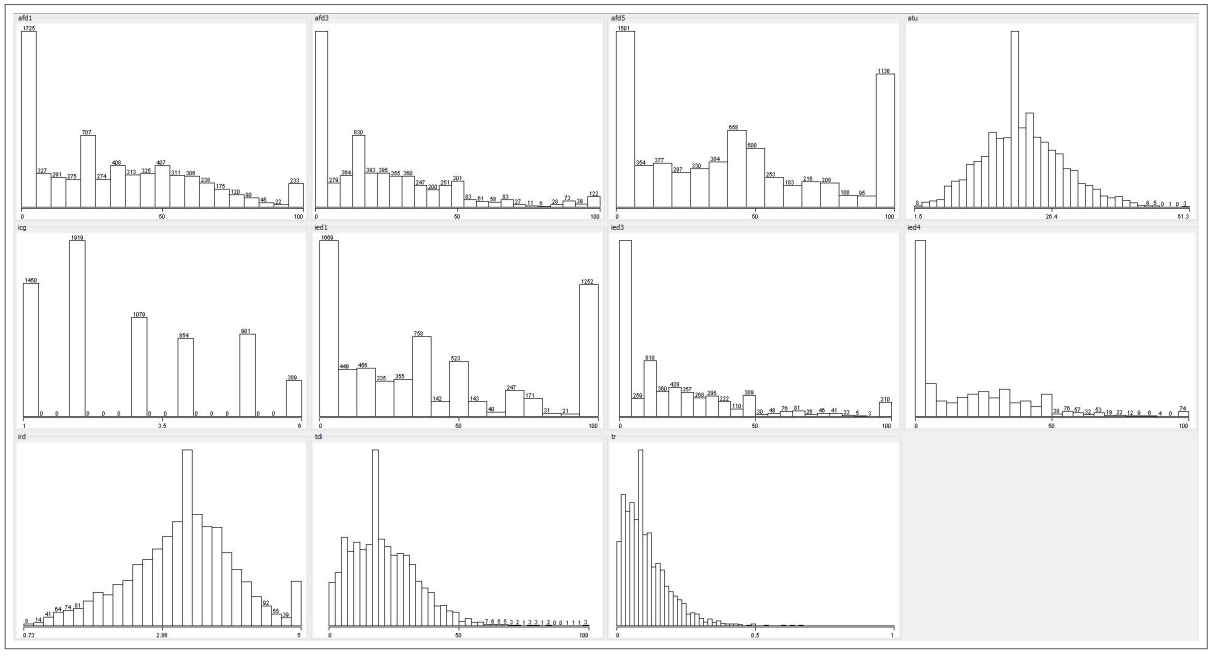


Figura 61 – Distribuição dos dados do conjunto de taxas de reprovação do ensino fundamental.

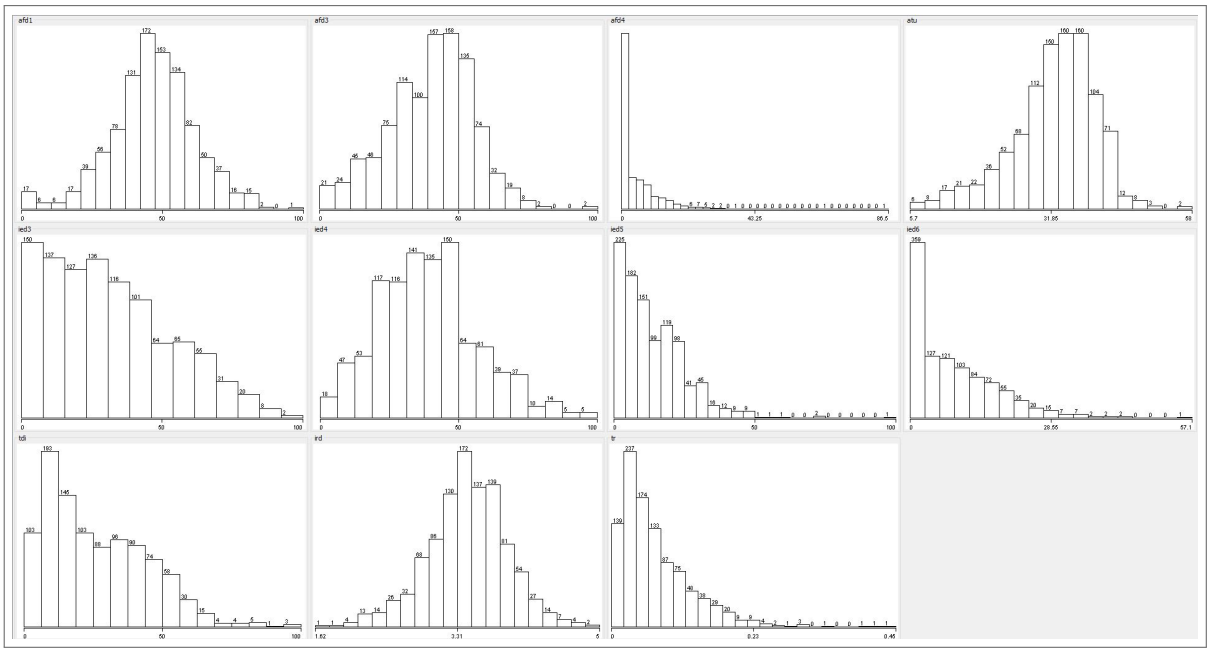


Figura 62 – Distribuição dos dados do conjunto de taxas de reprovação do ensino médio.

APÊNDICE B – CÓDIGO EM JAVA PARA USO DOS ALGORITMOS DA BIBLIOTECA DO WEKA 3.6.10

```
1
    /**
3    * Cria um classificador de acordo com o argumento especificado.
     *
5    * @param theClassifier: constante de classe referente ao classificador desejado
         .
     * @return Uma objeto do classificador.
7    */
    public Classifier createClassifier(int theClassifier) {
9
        if (theClassifier == J48) {
11          return new J48();
        } else if (theClassifier == J48_1) {
13          J48 j48 = new J48();
            j48.setUnpruned(true);
15          return j48;
        } else if (theClassifier == NAIVE_BAYES) {
17          return new NaiveBayes();
        } else if (theClassifier == NAIVE_BAYES_1) {
19          NaiveBayes nb = new NaiveBayes();
            nb.setUseKernelEstimator(true);
21          return nb;
        } else if (theClassifier == NAIVE_BAYES_2) {
23          NaiveBayes nb = new NaiveBayes();
            nb.setDisplayModelInOldFormat(true);
25          nb.setUseKernelEstimator(true);
            return nb;
27      } else if (theClassifier == RANDOM_FOREST) {
            return new RandomForest();
29      } else if (theClassifier == RANDOM_FOREST_1) {
            RandomForest randomForest = new RandomForest();
31          randomForest.setNumTrees(200);
            return randomForest;
33      } else if (theClassifier == ADABOOST) {
            AdaBoostM1 adaboost = new AdaBoostM1();
35          adaboost.setClassifier(new DecisionStump());
            return adaboost;
37      } else if (theClassifier == ADABOOST_1) {
            AdaBoostM1 adaboost = new AdaBoostM1();
39          adaboost.setClassifier(new REPTree());
            return adaboost;
41      } else if (theClassifier == ADABOOST_2) {
            AdaBoostM1 adaboost = new AdaBoostM1();
43          adaboost.setClassifier(new J48());
            return adaboost;
45      } else if (theClassifier == ADABOOST_3) {
            AdaBoostM1 adaboost = new AdaBoostM1();
47          adaboost.setClassifier(new LADTree());
            return adaboost;
49      } else if (theClassifier == ADABOOST_4) {
            AdaBoostM1 adaboost = new AdaBoostM1();
```

```
51              adaboost.setClassifier(new LMT());
                return adaboost;
53          } else if (theClassifier == ADABOOST_5) {
                AdaBoostM1 adaboost = new AdaBoostM1();
55              adaboost.setClassifier(new BFTree());
                return adaboost;
57          } else if (theClassifier == LOGISTIC_REGRESSION) {
                return new Logistic();
59          } else if (theClassifier == ONER){
                return new OneR();
61          } else if (theClassifier == SVM) {
                LibSVM libSVM = new LibSVM();
63              libSVM.setNormalize(true);
                return libSVM;
65          }  else if (theClassifier == SVM_1) {
                LibSVM libSVM = new LibSVM();
67              libSVM.setNormalize(true);
                libSVM.setCost(0.5);
69              return libSVM;
            } else if (theClassifier == SVM_2) {
71              LibSVM libSVM = new LibSVM();
                libSVM.setNormalize(true);
73              libSVM.setCost(0.1);
                return libSVM;
75          } else if (theClassifier == SVM_3) {
                LibSVM libSVM = new LibSVM();
77              libSVM.setNormalize(true);
                libSVM.setKernelType(new SelectedTag(LibSVM.KERNELTYPE_SIGMOID, LibSVM.
                    TAGS_KERNELTYPE));
79              return libSVM;
            }  else if (theClassifier == SVM_4) {
81              LibSVM libSVM = new LibSVM();
                libSVM.setNormalize(true);
83              libSVM.setCost(0.5);
                libSVM.setKernelType(new SelectedTag(LibSVM.KERNELTYPE_SIGMOID, LibSVM.
                    TAGS_KERNELTYPE));
85              return libSVM;
            } else if (theClassifier == SVM_5) {
87              LibSVM libSVM = new LibSVM();
                libSVM.setNormalize(true);
89              libSVM.setCost(0.1);
                libSVM.setKernelType(new SelectedTag(LibSVM.KERNELTYPE_SIGMOID, LibSVM.
                    TAGS_KERNELTYPE));
91              return libSVM;
            } else if (theClassifier == KNN) {
93              return new IBk();
            } else if (theClassifier == KNN_1) {
95              IBk ibk = new IBk(3);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_NONE, IBk.
                    TAGS_WEIGHTING));
97              return ibk;
            } else if (theClassifier == KNN_2) {
99              IBk ibk = new IBk(3);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_INVERSE, IBk.
                    TAGS_WEIGHTING));
101             return ibk;
            }else if (theClassifier == KNN_3) {
```

```
103             IBk ibk = new IBk(3);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_SIMILARITY, IBk.
                    TAGS_WEIGHTING));
105             return ibk;
            }else if (theClassifier == KNN_4) {
107             IBk ibk = new IBk(5);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_NONE, IBk.
                    TAGS_WEIGHTING));
109             return ibk;
            }else if (theClassifier == KNN_5) {
111             IBk ibk = new IBk(5);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_INVERSE, IBk.
                    TAGS_WEIGHTING));
113             return ibk;
            }else if (theClassifier == KNN_6) {
115             IBk ibk = new IBk(5);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_SIMILARITY, IBk.
                    TAGS_WEIGHTING));
117             return ibk;
            }else if (theClassifier == KNN_7) {
119             IBk ibk = new IBk(7);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_NONE, IBk.
                    TAGS_WEIGHTING));
121             return ibk;
            }else if (theClassifier == KNN_8) {
123             IBk ibk = new IBk(7);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_INVERSE, IBk.
                    TAGS_WEIGHTING));
125             return ibk;
            }else if (theClassifier == KNN_9) {
127             IBk ibk = new IBk(7);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_SIMILARITY, IBk.
                    TAGS_WEIGHTING));
129             return ibk;
            }else if (theClassifier == MLP) {
131             MultilayerPerceptron mlp = new MultilayerPerceptron();
                return mlp;
133         }else if (theClassifier == MLP_1) {
                MultilayerPerceptron mlp = new MultilayerPerceptron();
135             mlp.setLearningRate(0.1);
                return mlp;
137         } else if (theClassifier == MLP_2) {
                MultilayerPerceptron mlp = new MultilayerPerceptron();
139             mlp.setLearningRate(0.05);
                return mlp;
141         } else if (theClassifier == MLP_3) {
                MultilayerPerceptron mlp = new MultilayerPerceptron();
143             mlp.setHiddenLayers("a,2");
                return mlp;
145         } else if (theClassifier == MLP_4) {
                MultilayerPerceptron mlp = new MultilayerPerceptron();
147             mlp.setLearningRate(0.1);
                mlp.setHiddenLayers("a,2");
149             return mlp;
            } else if (theClassifier == MLP_5) {
151             MultilayerPerceptron mlp = new MultilayerPerceptron();
                mlp.setLearningRate(0.05);
```

```
153             mlp.setHiddenLayers("a,2");
                return mlp;
155         } else if (theClassifier == ZEROR) {
                ZeroR zeror = new ZeroR();
157             return zeror;
            } else if (theClassifier == STACKING) {
159             Stacking stacking = new Stacking();
                return stacking;
161         } else if (theClassifier == LINEAR_REGRESSION) {
                LinearRegression lr = new LinearRegression();
163             return lr;
            }  else if (theClassifier == LINEAR_REGRESSION_1) {
165             LinearRegression lr = new LinearRegression();
                lr.setAttributeSelectionMethod(new SelectedTag(LinearRegression.
                    SELECTION_GREEDY, LinearRegression.TAGS_SELECTION));
167             return lr;
            }  else if (theClassifier == GP) {
169             GaussianProcesses gp = new GaussianProcesses();
                return gp;
171         }  else if (theClassifier == GP_1) {
                GaussianProcesses gp = new GaussianProcesses();
173             gp.setFilterType(new SelectedTag(GaussianProcesses.FILTER_NONE,
                    GaussianProcesses.TAGS_FILTER));
                return gp;
175         }  else if (theClassifier == GP_2) {
                GaussianProcesses gp = new GaussianProcesses();
177             gp.setFilterType(new SelectedTag(GaussianProcesses.FILTER_STANDARDIZE,
                    GaussianProcesses.TAGS_FILTER));
                return gp;
179         }  else if (theClassifier == GP_3) {
                GaussianProcesses gp = new GaussianProcesses();
181             gp.setKernel(new Puk());
                return gp;
183         }  else if (theClassifier == GP_4) {
                GaussianProcesses gp = new GaussianProcesses();
185             gp.setFilterType(new SelectedTag(GaussianProcesses.FILTER_NONE,
                    GaussianProcesses.TAGS_FILTER));
                gp.setKernel(new Puk());
187             return gp;
            }  else if (theClassifier == GP_5) {
189             GaussianProcesses gp = new GaussianProcesses();
                gp.setFilterType(new SelectedTag(GaussianProcesses.FILTER_STANDARDIZE,
                    GaussianProcesses.TAGS_FILTER));
191             gp.setKernel(new Puk());
                return gp;
193         }  else if (theClassifier == M5R) {
                M5Rules m5R = new M5Rules();
195             return m5R;
            }  else if (theClassifier == M5R_1) {
197             M5Rules m5R = new M5Rules();
                m5R.setMinNumInstances(8);
199             return m5R;
            }  else if (theClassifier == M5R_2) {
201             M5Rules m5R = new M5Rules();
                m5R.setMinNumInstances(2);
203             return m5R;
            }  else if (theClassifier == M5R_3) {
```

```
205             M5Rules m5R = new M5Rules();
                m5R.setUnpruned(true);
207             return m5R;
            }  else if (theClassifier == M5R_4) {
209             M5Rules m5R = new M5Rules();
                m5R.setUnpruned(true);
211             m5R.setMinNumInstances(8);
                return m5R;
213         }  else if (theClassifier == M5R_5) {
                M5Rules m5R = new M5Rules();
215             m5R.setUnpruned(true);
                m5R.setMinNumInstances(2);
217             return m5R;
            }  else if (theClassifier == KSTAR) {
219             KStar kStar = new KStar();
                SelectedTag newMode = new SelectedTag(KStar.M_DELETE, KStar.TAGS_MISSING
                    );
221             kStar.setMissingMode(newMode);
                return kStar;
223         }  else if (theClassifier == ADDITIVE_REGRESSION) {
                AdditiveRegression additiveRegression = new AdditiveRegression();
225             additiveRegression.setClassifier(new REPTree());
                return additiveRegression;
227         }   else if (theClassifier == SUPPORT_VECTOR_REGRESSION) {
                SMOreg supportVectorRegression = new SMOreg();
229             return supportVectorRegression;
            }   else if (theClassifier == SUPPORT_VECTOR_REGRESSION_1) {
231             SMOreg supportVectorRegression = new SMOreg();
                supportVectorRegression.setC(0.1);
233             return supportVectorRegression;
            }   else if (theClassifier == SUPPORT_VECTOR_REGRESSION_2) {
235             SMOreg supportVectorRegression = new SMOreg();
                supportVectorRegression.setC(0.05);
237             return supportVectorRegression;
            }   else if (theClassifier == SUPPORT_VECTOR_REGRESSION_3) {
239             SMOreg supportVectorRegression = new SMOreg();
                supportVectorRegression.setKernel(new RBFKernel());
241             return supportVectorRegression;
            }   else if (theClassifier == SUPPORT_VECTOR_REGRESSION_4) {
243             SMOreg supportVectorRegression = new SMOreg();
                supportVectorRegression.setC(0.1);
245             supportVectorRegression.setKernel(new RBFKernel());
                return supportVectorRegression;
247         }   else if (theClassifier == SUPPORT_VECTOR_REGRESSION_5) {
                SMOreg supportVectorRegression = new SMOreg();
249             supportVectorRegression.setC(0.05);
                supportVectorRegression.setKernel(new RBFKernel());
251             return supportVectorRegression;
            }   else if (theClassifier == DECISION_TABLE) {
253             DecisionTable decisionTable = new DecisionTable();
                return decisionTable;
255         }   else if (theClassifier == DECISION_TABLE_1) {
                DecisionTable decisionTable = new DecisionTable();
257             decisionTable.setSearch(new GeneticSearch());
                return decisionTable;
259         }  else if (theClassifier == DECISION_TABLE_2) {
                DecisionTable decisionTable = new DecisionTable();
```

```
261             decisionTable.setSearch(new GreedyStepwise());
                return decisionTable;
263         }  else if (theClassifier == DECISION_TABLE_3) {
                DecisionTable decisionTable = new DecisionTable();
265             decisionTable.setEvaluationMeasure(new SelectedTag(DecisionTable.
                    EVAL_MAE, DecisionTable.TAGS_EVALUATION));
                return decisionTable;
267         }  else if (theClassifier == DECISION_TABLE_4) {
                DecisionTable decisionTable = new DecisionTable();
269             decisionTable.setSearch(new GeneticSearch());
                decisionTable.setEvaluationMeasure(new SelectedTag(DecisionTable.
                    EVAL_MAE, DecisionTable.TAGS_EVALUATION));
271             return decisionTable;
            }  else if (theClassifier == DECISION_TABLE_5) {
273             DecisionTable decisionTable = new DecisionTable();
                decisionTable.setSearch(new GreedyStepwise());
275             decisionTable.setEvaluationMeasure(new SelectedTag(DecisionTable.
                    EVAL_MAE, DecisionTable.TAGS_EVALUATION));
                return decisionTable;
277         }  else if (theClassifier == REP_TREE) {
                REPTree repTree = new REPTree();
279             return repTree;
            }  else if (theClassifier == REP_TREE_1) {
281             REPTree repTree = new REPTree();
                repTree.setMinNum(3);
283             return repTree;
            }  else if (theClassifier == REP_TREE_2) {
285             REPTree repTree = new REPTree();
                repTree.setMinNum(1);
287             return repTree;
            }  else if (theClassifier == REP_TREE_3) {
289             REPTree repTree = new REPTree();
                repTree.setNoPruning(true);
291             return repTree;
            }  else if (theClassifier == REP_TREE_4) {
293             REPTree repTree = new REPTree();
                repTree.setNoPruning(true);
295             repTree.setMinNum(3);
                return repTree;
297         }  else if (theClassifier == REP_TREE_5) {
                REPTree repTree = new REPTree();
299             repTree.setNoPruning(true);
                repTree.setMinNum(1);
301             return repTree;
            }  else if (theClassifier == CONJUNCTIVE_RULES) {
303             ConjunctiveRule conjunctiveRules = new ConjunctiveRule();
                return conjunctiveRules;
305         }  else if (theClassifier == CONJUNCTIVE_RULES_1) {
                ConjunctiveRule conjunctiveRules = new ConjunctiveRule();
307             conjunctiveRules.setMinNo(3);
                return conjunctiveRules;
309         }  else if (theClassifier == CONJUNCTIVE_RULES_2) {
                ConjunctiveRule conjunctiveRules = new ConjunctiveRule();
311             conjunctiveRules.setMinNo(5);
                return conjunctiveRules;
313         }  else if (theClassifier == CONJUNCTIVE_RULES_3) {
                ConjunctiveRule conjunctiveRules = new ConjunctiveRule();
```

```
315             conjunctiveRules.setMinNo(7);
                return conjunctiveRules;
317         }  else if (theClassifier == CONJUNCTIVE_RULES_4) {
                ConjunctiveRule conjunctiveRules = new ConjunctiveRule();
319             conjunctiveRules.setExclusive(true);
                conjunctiveRules.setMinNo(3);
321             return conjunctiveRules;
            }  else if (theClassifier == CONJUNCTIVE_RULES_5) {
323             ConjunctiveRule conjunctiveRules = new ConjunctiveRule();
                conjunctiveRules.setExclusive(true);
325             conjunctiveRules.setMinNo(5);
                return conjunctiveRules;
327         }  else if (theClassifier == CONJUNCTIVE_RULES_6) {
                ConjunctiveRule conjunctiveRules = new ConjunctiveRule();
329             conjunctiveRules.setExclusive(true);
                conjunctiveRules.setMinNo(7);
331             return conjunctiveRules;
            }  else if (theClassifier == RBF_NETWORK) {
333             RBFNetwork rbfNetwork = new RBFNetwork();
                return rbfNetwork;
335         }  else if (theClassifier == RBF_NETWORK_1) {
                RBFNetwork rbfNetwork = new RBFNetwork();
337             rbfNetwork.setNumClusters(3);
                return rbfNetwork;
339         }  else if (theClassifier == BAGGING) {
                Bagging bagging = new Bagging();
341             bagging.setClassifier(new REPTree());
                return bagging;
343         }  else if (theClassifier == BAGGING_1) {
                Bagging bagging = new Bagging();
345             bagging.setClassifier(new DecisionStump());
                return bagging;
347         }  else if (theClassifier == BAGGING_2) {
                Bagging bagging = new Bagging();
349             bagging.setClassifier(new J48());
                return bagging;
351         }  else if (theClassifier == BAGGING_3) {
                Bagging bagging = new Bagging();
353             bagging.setClassifier(new LADTree());
                return bagging;
355         }  else if (theClassifier == BAGGING_4) {
                Bagging bagging = new Bagging();
357             bagging.setClassifier(new LMT());
                return bagging;
359         }  else if (theClassifier == BAGGING_5) {
                Bagging bagging = new Bagging();
361             bagging.setClassifier(new BFTree());
                return bagging;
363         }  else if (theClassifier == BAGGING_KNN_1) {
                Bagging bagging = new Bagging();
365             IBk ibk = new IBk(3);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_NONE, IBk.
                    TAGS_WEIGHTING));
367             bagging.setClassifier(ibk);
                return bagging;
369         }  else if (theClassifier == BAGGING_KNN_2) {
                Bagging bagging = new Bagging();
```

```
371             IBk ibk = new IBk(3);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_INVERSE, IBk.
                    TAGS_WEIGHTING));
373             bagging.setClassifier(ibk);
                return bagging;
375         }  else if (theClassifier == BAGGING_KNN_3) {
                Bagging bagging = new Bagging();
377             IBk ibk = new IBk(3);
                ibk.setDistanceWeighting(new SelectedTag(IBk.WEIGHT_SIMILARITY, IBk.
                    TAGS_WEIGHTING));
379             bagging.setClassifier(ibk);
                return bagging;
381         }  else if (theClassifier == BAGGING_MLP_1) {
                Bagging bagging = new Bagging();
383             MultilayerPerceptron mlp = new MultilayerPerceptron();
                mlp.setLearningRate(0.1);
385             bagging.setClassifier(mlp);
                return bagging;
387         }  else if (theClassifier == BAGGING_MLP_2) {
                Bagging bagging = new Bagging();
389             MultilayerPerceptron mlp = new MultilayerPerceptron();
                mlp.setLearningRate(0.05);
391             bagging.setClassifier(mlp);
                return bagging;
393         }  else if (theClassifier == BAGGING_MLP_3) {
                Bagging bagging = new Bagging();
395             MultilayerPerceptron mlp = new MultilayerPerceptron();
                mlp.setHiddenLayers("a,2");
397             bagging.setClassifier(mlp);
                return bagging;
399         }  else if (theClassifier == BAGGING_MLP_4) {
                Bagging bagging = new Bagging();
401             MultilayerPerceptron mlp = new MultilayerPerceptron();
                mlp.setHiddenLayers("a,2");
403             mlp.setLearningRate(0.1);
                bagging.setClassifier(mlp);
405             return bagging;
            }  else if (theClassifier == BAGGING_MLP_5) {
407             Bagging bagging = new Bagging();
                MultilayerPerceptron mlp = new MultilayerPerceptron();
409             mlp.setHiddenLayers("a,2");
                mlp.setLearningRate(0.05);
411             bagging.setClassifier(mlp);
                return bagging;
413         }  else if (theClassifier == BAGGING_KSTAR) {
                Bagging bagging = new Bagging();
415             KStar kStar = new KStar();
                SelectedTag newMode = new SelectedTag(KStar.M_DELETE, KStar.TAGS_MISSING
                    );
417             kStar.setMissingMode(newMode);
                bagging.setClassifier(kStar);
419             return bagging;
            }  else if (theClassifier == BAGGING_BAYES_NET) {
421             Bagging bagging = new Bagging();
                BayesNet bayesNet = new BayesNet();
423             bagging.setClassifier(bayesNet);
                return bagging;
```

```
425         }  else if (theClassifier == M5P) {
                M5P m5P = new M5P();
427             return m5P;
            }  else if (theClassifier == M5P_1) {
429             M5P m5P = new M5P();
                m5P.setMinNumInstances(8);
431             return m5P;
            }  else if (theClassifier == M5P_2) {
433             M5P m5P = new M5P();
                m5P.setMinNumInstances(2);
435             return m5P;
            }  else if (theClassifier == M5P_3) {
437             M5P m5P = new M5P();
                m5P.setUnpruned(true);
439             return m5P;
            }  else if (theClassifier == M5P_4) {
441             M5P m5P = new M5P();
                m5P.setMinNumInstances(8);
443             m5P.setUnpruned(true);
                return m5P;
445         }  else if (theClassifier == M5P_5) {
                M5P m5P = new M5P();
447             m5P.setMinNumInstances(2);
                m5P.setUnpruned(true);
449             return m5P;
            }  else if (theClassifier == DECISION_STUMP) {
451             DecisionStump decisionStump = new DecisionStump();
                return decisionStump;
453         }  else if (theClassifier == LEAST_MED_SQ) {
                LeastMedSq leastMedSq = new LeastMedSq();
455             return leastMedSq;
            }  else if (theClassifier == LEAST_MED_SQ_1) {
457             LeastMedSq leastMedSq = new LeastMedSq();
                leastMedSq.setSampleSize(5);
459             return leastMedSq;
            }  else if (theClassifier == LEAST_MED_SQ_2) {
461             LeastMedSq leastMedSq = new LeastMedSq();
                leastMedSq.setSampleSize(6);
463             return leastMedSq;
            }  else if (theClassifier == LEAST_MED_SQ_3) {
465             LeastMedSq leastMedSq = new LeastMedSq();
                leastMedSq.setRandomSeed(1);
467             return leastMedSq;
            }  else if (theClassifier == LEAST_MED_SQ_4) {
469             LeastMedSq leastMedSq = new LeastMedSq();
                leastMedSq.setRandomSeed(1);
471             leastMedSq.setSampleSize(5);
                return leastMedSq;
473         }  else if (theClassifier == LEAST_MED_SQ_5) {
                LeastMedSq leastMedSq = new LeastMedSq();
475             leastMedSq.setRandomSeed(1);
                leastMedSq.setSampleSize(6);
477             return leastMedSq;
            }  else if (theClassifier == LOCALLY_WEIGHTED_LEARNING) {
479             LWL lwl = new LWL();
                return lwl;
481         }  else if (theClassifier == LOCALLY_WEIGHTED_LEARNING_1) {
```

```
            LWL lwl = new LWL();
483         LinearNNSearch lns = new LinearNNSearch();
            try {
485             lns.setDistanceFunction(new ManhattanDistance());
            } catch (Exception e) {
487             e.printStackTrace();
            }
489         lwl.setNearestNeighbourSearchAlgorithm(lns);
            return lwl;
491     }  else if (theClassifier == BAYES_NET) {
            BayesNet bayesNet = new BayesNet();
493         return bayesNet;
        }  else if (theClassifier == LMT) {
495         LMT lmt = new LMT();
            return lmt;
497     }  else if (theClassifier == LAD_TREE) {
            LADTree ladTree = new LADTree();
499         return ladTree;
        } else if (theClassifier == J_RIP) {
501         JRip jRip = new JRip();
            return jRip;
503     }   else if (theClassifier == BEST_FIRST_TREE) {
            BFTree bestFirstTree = new BFTree();
505         bestFirstTree.setPruningStrategy(new SelectedTag(BFTree.
                PRUNING_PREPRUNING, BFTree.TAGS_PRUNING));
            return bestFirstTree;
507     }  else if (theClassifier == BEST_FIRST_TREE_1) {
            BFTree bestFirstTree = new BFTree();
509         bestFirstTree.setPruningStrategy(new SelectedTag(BFTree.PRUNING_UNPRUNED
                , BFTree.TAGS_PRUNING));
            return bestFirstTree;
511     }  else if (theClassifier == BEST_FIRST_TREE_2) {
            BFTree bestFirstTree = new BFTree();
513         return bestFirstTree;
        }
515

517     else {
            return new J48();
519     }
    }
```