# An exploratory study about the cross-project defect prediction: impact of using different classification algorithms and a measure of performance in building predictive models.

Ricardo F. P. Satin*, Igor Scaliante Wiese†, Reginaldo Ré†
Departamento Acadêmico de Computação
Universidade Tecnológica Federal do Paraná – Campo Mourão
Paraná, Brasil
Email: ricardo.francisco@gmail.com*, {igor, reginaldo}@utfpr.edu.br†

***Abstract*—Predicting defects in software projects is a complex task, especially in the initial phases of software development because there are a few available data. The use of cross-project defect prediction is indicated in such situation because it enables to reuse data of similar projects. In order to find and group similar projects, this paper proposes the construction of cross-project prediction models using a measure of performance achieved through the application of classification algorithms. To do so, we studied the combined application of different algorithms of classification, of feature selection, and clustering data, applied to 1270 projects aiming to building different cross-project prediction models. In this study we concluded that Naive Bayes algorithm obtained the best performance, with 31.58 % of satisfactory predictions in 19 models created with its use. This proposal seems to be promise, once the local predictions considered satisfactory reached 31.58%, against 26.31 % of global predictions.**

***Index-terms*—Software maintenance, software quality, defect prediction models, cross-project defect prediction models.**

## I. Introdução

O processo de predição de defeitos busca predizer a existência de defeitos antes que eles efetivamente ocorram [1], [2]. O uso de seus resultados permite direcionar esforços das equipes prioritariamente em diferentes tarefas, para, por exemplo: minimizar o tempo gasto com a correção de defeitos; reduzir os custos de manutenção; e, incrementar a percepção de qualidade final do produto [1], [3].

Contudo, predizer defeitos que ocorrerão em um projeto de software é uma tarefa complexa, especialmente nas fases iniciais do desenvolvimento [4]. Nessas fases, geralmente, tais projetos não têm dados históricos suficientes para que sejam construídos modelos próprios de predição com o uso, por exemplo, dados de versões anteriores. Para resolver esse problema a predição cruzada de defeitos entre projetos (do inglês, *cross-project defect prediction*) pode ser utilizada, de maneira que projetos sem dados históricos suficientes usem dados de projetos semelhantes, e assim os modelos de predição possam ser transferidos entre projetos [1], [5], [6]. Portanto, é importante a pesquisa de métodos que permitam a identificação de características semelhantes entre projetos distintos para que a predição cruzada entre eles seja aplicável [7].

O uso de algoritmos de agrupamento permite encontrar padrões de similaridade entre projetos. Isso pode ser feito por meio de métricas disponíveis que permitam a comparação entre eles para estabelecer relações entre os projetos. Essa é uma das técnicas mais utilizadas em pesquisas anteriores que visam a predição de defeitos entre projetos [4], [5], [7], [8]. Tais pesquisas tentam criar modelos de predição cruzada genéricos, que poderiam ser usados para predizer defeitos em qualquer projeto alvo, desde que as técnicas empregadas pelos modelos encontrem semelhanças suficientes entre os projetos usados na construção do modelo e o projeto alvo [4], [5], [7].

As propostas de modelos de predição cruzada de defeitos entre projetos encontradas na literatura usam métricas dos projetos para encontrar semelhanças. Ao contrário de tais estudos, este trabalho propõe uma forma alternativa de construção de modelos para a predição cruzada baseada em agrupamentos de projetos por similaridade. Propõem-se a utilização de um algoritmo de agrupamento por similaridade chamado BSAS (do inglês, *Basic Sequencial Algorithmic Scheme*) para construir modelos de predição que utilizam uma medida de desempenho de algoritmos de classificação denominada AUC (do inglês, *Area Under a Curve*). Para viabilizar a presente proposta, as seguintes questões de pesquisa são respondidas:

- QP1: Dentre 5 algoritmos de classificação escolhidos na literatura, algum apresenta desempenho melhor para ser utilizado com a técnica proposta? Para responder essa questão de pesquisa, foram comparados 5 algoritmos de classificação em 15 diferentes cenários aplicados a 1270 projetos de software. Os resultados encontrados indicam que o algoritmo *Naive Bayes* foi ligeiramente melhor que os algoritmos *Simple Logistic*, *Random Forest*, *Decision Table*, e J48. No entanto, com o uso de teste estatístico de hipótese não foi possível comprovar o melhor desempenho do *Naive Bayes* frente aos outros 4 algoritmos.
  - QP1.1: Dentre os 3 métodos selecionadores de atri-

978-1-4673-9143-6/15/$31.00 

butos escolhidos na literatura, algum apresenta desempenho melhor para ser utilizado com a técnica proposta? Para responder essa questão de pesquisa, foram comparados 3 algoritmos de classificação em 15 diferentes cenários aplicados a 1270 projetos de software. Os resultados indicam que existe diferença estatística que apontam como melhor método o par CFS/*Genetic Search*.

- QP2: Existe diferença nos resultados da predição cruzada de defeitos quando a técnica proposta neste trabalho é utilizada? Sim. Foi verificado com a implementação da técnica que a combinação do algoritmo de agrupamento BSAS com o uso dos valores de AUC obtido com a execução da predição cruzada, combinada com os algoritmos de seleção de atributos, que as predições locais, consideradas satisfatórias, totalizaram 31,58% dos resultados encontrados, contra 26,31% das predições globais. Contudo, em alguns casos os resultados não geraram bons agrupamentos quando projetos têm a grande maioria de instâncias anotadas como defeitos ou livres de defeitos.

O uso do agrupamento por valores de AUC, na maioria dos casos, os projetos foram agrupados de forma que o desempenho da predição de defeitos produza um resultado do próprio AUC maior que valores obtidos por um classificador randômico. Esse resultado é promissor, uma vez que não foram encontrados na literatura trabalhos que usam uma medida de desempenho para agrupar projetos similares, e possibilita explorar novas alternativas para predição cruzada de defeitos. Pode-se citar 2 principais contribuições deste trabalho:

- os modelos de predição cruzada entre projetos construídos com o auxílio de agrupamento de projetos similares pelo valor de AUC; e,
- um estudo sobre o impacto de diferentes classificadores, tanto no processo de predição quanto no agrupamento de projetos similares.

O restante deste artigo está estruturado da seguinte forma: na Seção II são apresentados os passos de construção dos modelos de predição cruzada de defeitos apoiada por similaridade entre projetos; os resultados obtidos nos passos de construção dos modelos, bem como detalhes sobre as questões de pesquisa são discutidos na Seção III; já, nas Seções IV, V e VI são descritas, respectivamente, algumas limitações da presente proposta, algumas das principais pesquisas relacionadas com tema deste trabalho, e algumas sugestões de direções da presente pesquisa; por fim, as principais conclusões são descritas na Seção VII.

## II. Modelos de predição cruzada de defeitos apoiada por similaridade de projetos

### A. Criação do modelo de predição cruzada de defeitos

A predição de defeitos pode ser obtida por meio da aplicação de modelos de predição em projetos nos quais se pretende avaliar a presença ou ausência de defeitos. Segundo Hall *et al.* [1], um modelo de predição de defeitos é construído seguindo critérios de uma metodologia que permita seu desenvolvimento, treino, teste e avaliação de desempenho. Procurou-se seguir esses passos neste trabalho. No entanto, como a presente proposta visa propor uma técnica de predição cruzada de projetos que utiliza similaridade entre eles, tais passos sofreram modificações, conforme ilustrado na Figura 1, que mostra o processo por meio do qual os modelos de predição cruzada de defeitos foram desenvolvidos, treinados, testados e tiveram seu desempenho avaliado.

### B. Tratar dados de projetos

Neste trabalho, o conjunto de dados de projetos (do inglês, *dataset*) utilizado foi o mesmo proposto por Zhang *et al.* [7]. Ele tem 1398 projetos de software livre com 72 atributos, compostas predominantemente por métricas de código. Uma dessas métricas é a variável dependente do modelo de predição. Essa variável é dicotômica e pode assumir o valor *Buggy* ou *Clean*. Essas duas classes, *Buggy* ou *Clean*, indicam respectivamente a presença ou ausência de defeitos em um dado arquivo – ou instância – dos projetos.

Muito embora o conjunto de dados tenha que passar pelos tratamentos adequados, não se pode garantir sua qualidade. A qualidade dos dados depende basicamente de atividades que antecedem o próprio processo de mineração e da construção do conjunto de dados, fases que não são atacadas neste trabalho. Assim, o problema da qualidade dos dados pode influenciar no desempenho dos classificadores, e é discutido por alguns estudos [9], [10]. Outro ponto que vale ressaltar é que existem outros conjuntos de dados disponíveis na literatura especializada, como os dados disponíveis no repositório PROMISE[1] e NASA MDP[2]. No entanto, mesmo com essa disponibilidade, existem trabalhos que mostram que não se pode garantir a qualidade desses conjuntos de dados também [2], [4], [9]. Além disso, de maneira geral, os conjuntos de dados disponíveis possuem poucos projetos, por exemplo, o conjunto de dados do PROMISE, tem 76 projetos em sua maior versão. O conjunto de dados usado neste trabalho foi escolhido justamente por possuir um número muito maior de projetos e também por não ser apontado na literatura especializada como sendo um conjunto de dados com baixa qualidade.

Além da escolha de um conjunto de dados confiável, faz-se necessário trabalhar os dados de maneira que eles fiquem adequados as necessidades da construção dos modelos de predição. O objetivo da atividade mostrada na Figura 1, "1 Tratar Dados de Projetos", é produzir um conjunto consistente e minimamente aceitável de dados de projetos a partir de um conjunto de dados não tratados. Essa é uma etapa importante, que precede a utilização do conjunto de dados nos modelos de predição. Nela, um especialista utiliza seu conhecimento para tratar métricas e dados de maneira adequada. Além disso, o tratamento deve seguir, segundo Faceli *et al.* [11], um conjunto de práticas de preparação, tendo, eventualmente,

[1] http://openscience.us/repo/

[2] http://nasa-softwaredefectdatasets.wikispaces.com

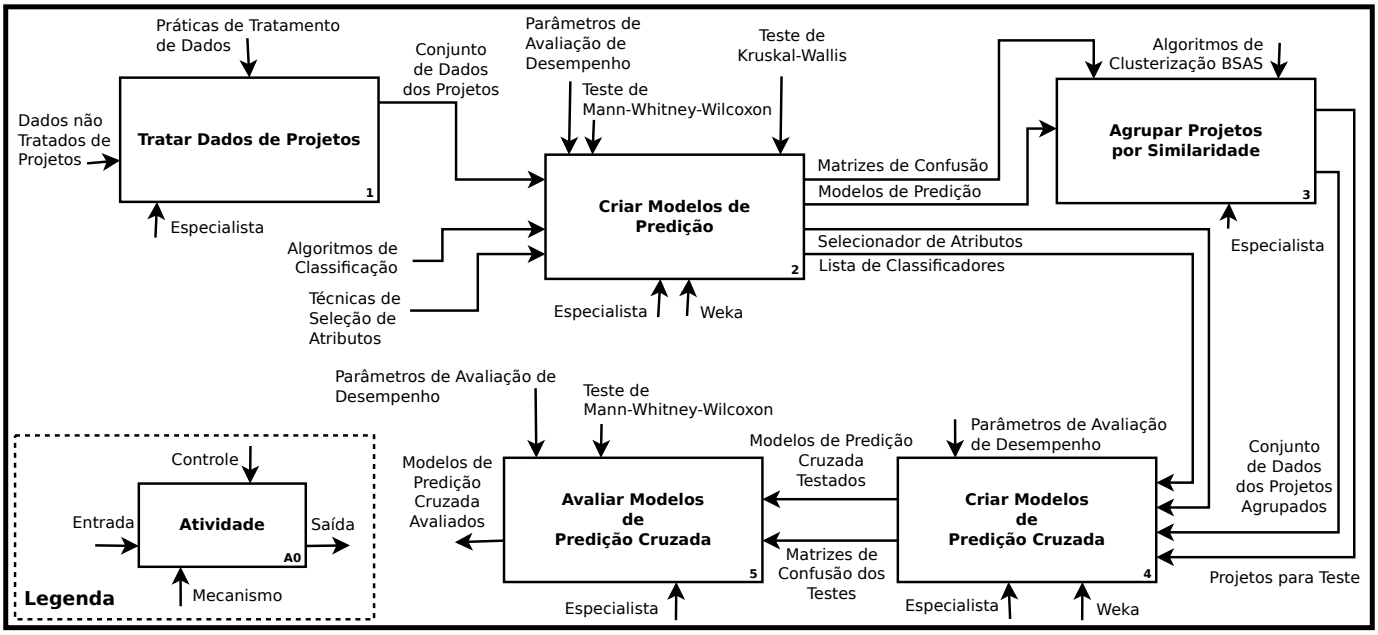


Fig. 1. Atividades de criação dos modelos de predição cruzada de defeitos com o uso de similaridade de projetos.

atributos relevantes mantidos, dados balanceados, dimensões reduzidas e transformadas.

Com o tratamento feito nos dados dos projetos, os 72 atributos originais foram reduzidas para 67, e os projetos, foram reduzidos de 1398 para 1270. Projetos com menos de 10 instâncias foram descartados, porque a técnica de treino de modelos usada na atividade "2 Criar Modelos de Predição", exige um mínimo de 10 instâncias. Ademais, atributos que não colaboram para o processo de predição foram removidas. Um exemplo de atributo descartado foi "origem dos projetos", um atributo nominal que armazena de onde o projeto foi minerado, por exemplo, *Source Forge*[3] ou *Google Code*[4].

### C. Criar modelos de predição

Na Figura 2 são apresentadas as subatividades para a criação dos modelos de predição dos projetos, que detalha a atividade mostrada na Figura 1, "2 Criar Modelos de Predição". De forma geral, as atividades que detalham a criação dos modelos de predição seguem a proposta de Hall *et al.* [1]: desenvolvimento, treino, teste e avaliação de desempenho. O objetivo da atividade "2 Criar Modelos de Predição" é produzir: uma lista de classificadores que se desejou avaliar, conseguido a partir de algoritmos de classificação encontrados na literatura especializada; um selecionador de atributos que trabalhe em conjunto com os classificadores selecionados, escolhido a partir de técnicas de seleção de atributos, também encontrados na literatura especializada; e a matriz de confusão, produzida a partir do treino e teste dos modelos de predição de defeitos de cada um dos projetos. Um especialista utilizou a ferramenta Weka[5], para trabalhar com as entradas dessa atividade e produzir os resultados esperados. Os detalhes de como isso foram feitos são apresentados nas subatividades descritas a seguir.

Na primeira das subatividades, "2.1 Escolher Classificadores", foram escolhidos os classificadores para compor a lista de algoritmos de classificação. A escolha de classificadores adequados ainda é um assunto em aberto quando se trata de construção de modelos de predição de defeitos. Estudos são controversos sobre a influência dos algoritmos de classificação no desempenho obtido pelo processo de predição [9], [10]. Lessman *et al.* [10] aponta em seu trabalho que não existem diferenças estatísticas relevantes nos resultados gerados pelo classificador utilizado no processo de predição. No entanto, em um estudo mais recente, essa afirmação é refutada por Ghotra *et al.* [9], que apontou problemas na qualidade do conjunto de dados dos projetos, e consequentemente das métricas, usados no experimento conduzido por Lessman *et al.*, o que pode ter influenciado nos resultados obtidos.

Assim sendo, dentre os classificadores mais utilizados em trabalhos similares [4], [5], [12], e com os trabalhos de Ghotra *et al.* [9] e Lessman *et al.* [10] em consideração, procurou-se selecionar algoritmos com bom desempenho, mas que utilizam abordagens distintas. Isso foi feito com o intuito de avaliar a influência dos classificadores nos resultados do processo de predição, uma vez que esse assunto é controverso. Os algoritmos utilizados na construção dos modelos de predição foram: com atuação estatística, *Naive Bayes*, que tem a característica de assumir que todos os atributos são independentes entre si, e *Simple Logistic*, que usa regressão logística linear; baseados em árvores de decisão, J48, que usa estatística para a geração de árvores de decisão, e *Random*

[3]http://sourceforge.net/
[4]https://code.google.com/
[5]http://www.cs.waikato.ac.nz/ml/weka/

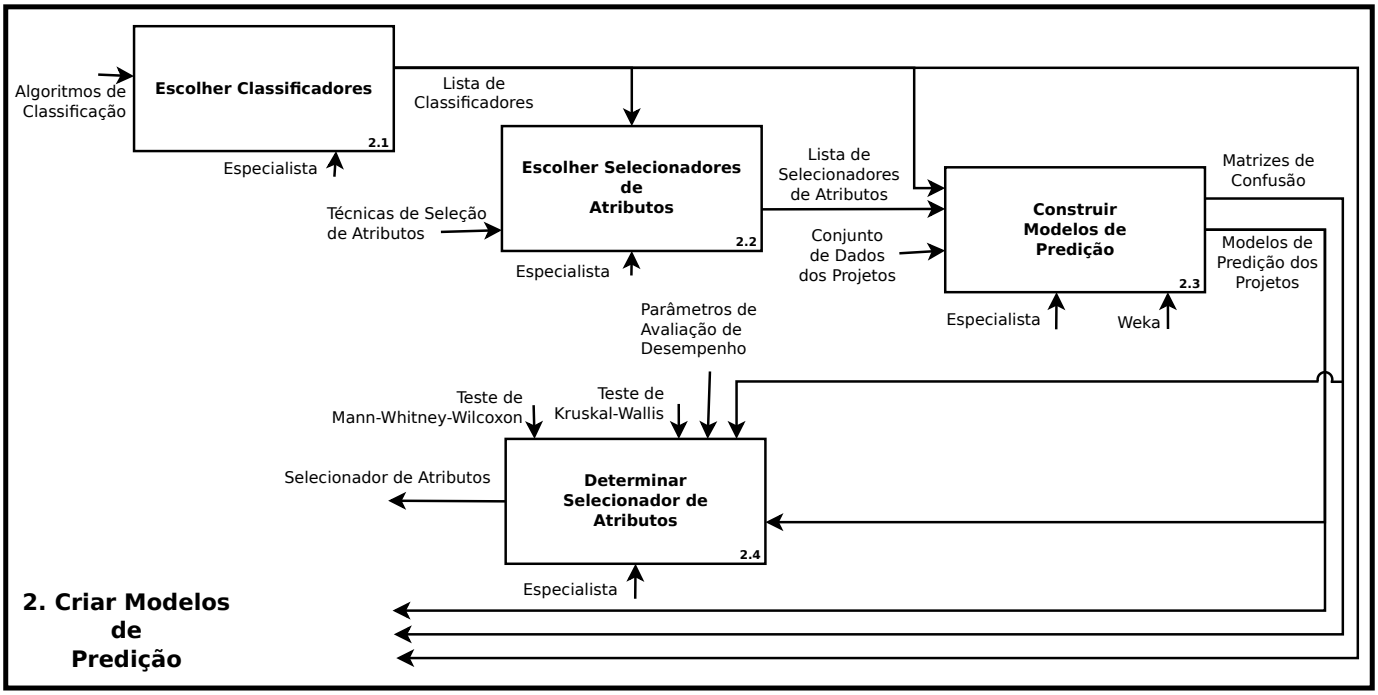


Fig. 2. Atividades de criação dos modelos de predição individual dos projetos.

*Forest*, que é um puro construtor de árvores de decisão; e, que usa regras de decisão, *Decision Table*, que modela um conjunto de regras complexas e suas ações correspondentes.

As entradas dos algoritmos de classificação são os atributos dos projetos contidos no conjunto de dados, também chamados de métricas. Na construção dos modelos de predição eles assumem o papel de preditores [13], [14]. Uma quantidade considerável de métricas está catalogada na literatura: existem métricas relacionadas a critérios técnicos do software, como a complexidade do projeto [15]. Também existem métricas relacionadas ao processo de construção do software, como o nível de experiência dos programadores [16]. Por fim, existem outras relacionadas a fatores sociais do processo de desenvolvimento, como o tipo e meio de comunicação utilizado pelos desenvolvedores [17]–[19]. Alguns autores avaliam que a escolha das métricas adequadas é que pode viabilizar a transferência de modelos de predição entre projetos [1]. Inicialmente as métricas são obtidas pela mineração de informações disponíveis nos projetos, depois são refinadas para que as mais relevantes sejam mantidas. Uma forma de fazer isso é com a utilização de informações de especialistas ou com a criação de algoritmos para buscar identificar padrões e correlacionar as métricas utilizadas entre projetos [4]. A subatividade seguinte, "2.2 Escolher Selecionadores de Atributos", visa a escolher técnicas para selecionar atributos, e assim permitir a diminuição da multidimensionalidade dos dados para melhorar os modelos de predição e viabilizar a transferência desses modelos de predição [1].

Neste trabalho, a opção foi a utilização de técnicas de seleção de atributos (do inglês, *feature selection*), que trabalham de maneira conjunta com os classificadores. Essas técnicas procuram encontrar entre todos os atributos do conjunto de dados aqueles que conseguem dar as melhores informações para os algoritmos de classificação. Uma técnica de seleção de atributos é constituída basicamente de dois tipos de algoritmos para realizar seu trabalho [20]. O primeiro tipo agrupa àqueles que estabelecem formas por meio dos quais subconjuntos de atributos são pesquisados e encontrados, chamados de métodos de pesquisa (do inglês, *Search Methods*). Além dos algoritmos de métodos de pesquisa, também são utilizados algoritmos que procuram estabelecer uma relação entre os atributos avaliados, chamados de avaliadores de atributos (do inglês, *Attribute Evaluators*).

Foi usado o avaliador de atributos CFS (do inglês, *Correlation-based Feature Selection*), por ser um dos mais utilizados por trabalhos anteriores [21]. O CFS é um algoritmo que considera os atributos de forma independente, que avalia a capacidade preditiva de cada um e o grau de redundância entre eles. Subconjuntos de atributos, que são altamente correlacionados com a classe que se pretende predizer são preferidos pelo algoritmo. Alguns métodos de pesquisa foram selecionados: *Best First*, que implementa uma pesquisa por conjuntos de atributos, que permite mudar a direção da busca sem perder as informações já encontradas até um dado momento; *Genetic Search*, que utiliza os princípios de seleção natural da biologia e, para isso, utiliza conceitos de mutação e cruzamento de valores; e, *Greedy Stepwise*, que percorre subconjuntos de atributos e busca características para melhorar o processo de predição, e para apenas quando não consegue melhorar os resultados encontrados.

Na subatividade seguinte, "2.3 Construir Modelos de Predição", os modelos de predição foram desenvolvidos,

TABELA I
CENÁRIOS PARA CRIAÇÃO DE MODELOS DE PREDIÇÃO.

| **Classificador** | **Técnica de Seleção de Atributos** | | |
|---|---|---|---|
| | **CFS** | | |
| | ***Best First*** | ***Genetic Search*** | ***Greedy Stepwise*** |
| *Naive Bayes* | Cenário 01 | Cenário 06 | Cenário 11 |
| *Random Forest* | Cenário 02 | Cenário 07 | Cenário 12 |
| *Decision Table* | Cenário 03 | Cenário 08 | Cenário 13 |
| *Simple Logistic* | Cenário 04 | Cenário 09 | Cenário 14 |
| J48 | Cenário 05 | Cenário 10 | Cenário 15 |

treinados e testados. Nesta atividade foram criados 15 cenários de desenvolvimento de modelos de predição, conforme pode ser visto na Tabela I. Cada um dos 15 cenários foi aplicado individualmente aos 1270 projetos, compondo assim os modelos de predição. Foram desenvolvidos, então, 19050 modelos de predição de defeitos. Na coluna "Classificador", da Tabela I, são exibidos os algoritmos de classificação utilizados. A coluna "Técnica de Seleção de Atributos" mostra os algoritmos de métodos de pesquisa e seus respectivos algoritmos de avaliação de atributos. A ideia dessa atividade foi permitir a avaliação dos classificadores e das técnicas de seleção de atributos nos diversos modelos de predição.

Os 19050 modelos foram treinados e testados com o uso da técnica de validação cruzada com a opção *10-fold cross-validation* [22]. A técnica promove a divisão da base de dados original em *"K"* partições idealmente estratificadas. Isto é, mantendo a proporção das classes da base de dado original. Após isso, o processo executa *"K"* interações aonde em cada interação é utilizado *"K-1"* conjuntos para treinamento e um conjunto para teste. Com isso, todas as partições são testadas [23]. Os valores obtidos para verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos foram coletados, com a formação da matriz de confusão de cada modelo. A matriz de confusão foi base para chegar ao valor de AUC, que representa a relação entre as taxas de falsos positivos e verdadeiros positivos dos modelos de predição calculado a partir da curva ROC (do inglês, *Receiver Operating Characteristic*) [24].

A atividade "2.4 Determinar Selecionador de Atributos" foi responsável por determinar o método de pesquisa, que trabalha junto com o avaliador de atributos, para juntos diminuir a dimensionalidade do conjunto de dados. A fim de averiguar se existe diferença de desempenho entre os 3 métodos de pesquisa escolhidos, o teste estatístico de Kruskal-Wallis [25] foi aplicado aos valores AUC obtidos a partir do teste dos projetos nos diferentes cenários de criação de modelos de predição. O valor obtido para $p$ foi de $2,2 * 10^{-16}$, menor que $0,05$, que indica que houve diferença estatística relevante entre as amostras. Para verificar qual dos 3 métodos de pesquisa obteve melhor desempenho, o teste estatístico não paramétrico para amostras independentes de Mann-Whitney-Wilcoxon [26] foi aplicado para *Genetic Search* e *BestFirst*. O resultado obtido foi $p = 2,2 * 10^{-16}$, menor que $0,05$, que indica que o algoritmo *Genetic Search* tem desempenho superior ao algoritmo *Best First*. Apenas para melhorar o entendimento dos resultados, foi calculada a frequência dos métodos de pesquisa que obtiveram valor de AUC $>= 0,7$ nos 19050 modelos de predição (4901 modelos). O resultado da contagem é apresentado na Tabela II, em que é possível observar que *Genetic Search* obteve melhor resultado, 1791 modelos de predição (36,54%), também na contagem de frequência.

TABELA II
FREQUÊNCIA DOS MÉTODOS DE PESQUISA QUE OBTIVERAM O MELHOR VALOR DE AUC NOS 19050 MODELOS DE PREDIÇÃO.

| **Método de pesquisa** | **Frequência** | **% Freq.** |
|---|---|---|
| *Genetic Search* | 1791 | 36,54% |
| *Best First* | 1575 | 32,13% |
| *Greedy Stepwise* | 1535 | 31,32% |
| Total | 4901 | |

### *D. Agrupar projetos por similaridade*

O objetivo desta atividade foi agrupar os projetos que são similares, segundo o critério proposto neste trabalho, e também criar uma lista dos projetos escolhidos para testar cada um dos modelos de predição cruzada na atividade seguinte. Cada agrupamento de projetos deu origem a um modelo de predição cruzada de defeitos[6].

Os valores de AUC calculados a partir das matrizes de confusão obtidas na atividade anterior constituem os parâmetros por meio do qual o agrupamento de projetos foi feito. O valor de AUC de cada um dos modelos não foi usado para avaliar individualmente os modelos, ao invés disso, ele foi usado para estabelecer a relação de similaridade entre os projetos para o processo de predição cruzada. Isso foi feito porque se esperava obter bons resultados dos modelos de predição cruzada com agrupamento de projetos, mesmo aqueles projetos com valores de AUC considerados ruins. Dessa maneira, a transferência de modelos de predição entre projetos seria beneficiada.

Para que o agrupamento propriamente dito fosse realizado, foi necessário selecionar um algoritmo de clusterização e, o elevado número de modelos de predição, foi um fator determinante na escolha. Isso porque, alguns algoritmos de clusterização, como por exemplo o K-means [27], precisam de um elemento centróide inicial como forma de inicializar o processo de agrupamento. Diferentes escolhas desse centróide inicial resultam em diferentes agrupamentos o que, consequentemente, pode levar a resultados não satisfatórios. Assim, não é uma tarefa trivial escolher um modelo de predição ótimo, que represente o centróide inicial, considerando o grande número de modelos de predição obtidos. Portanto, o elevado

[6]O termo "agrupamento" é usado quando se deseja referenciar os projetos e suas instâncias, contidos no conjunto de dados usado, agrupados segundo a aplicação de um método de agrupamento por similaridade. Já o termo "modelo de predição cruzada" é usado quando se deseja referenciar os projetos e suas instâncias já agrupados, juntamente com um algoritmo de classificação e um algoritmo selecionador de atributos. O termo "modelo de predição cruzada" é usado indistintamente para modelos não testados, testados, e testados e avaliados.

número de modelos de predição juntamente com a dificuldade de identificação de uma forma de inicializar os agrupamentos levou a escolha de um algoritmo de avaliação sequencial, que pudesse avaliar individualmente cada modelo [28]. Dentre esses algoritmos, um dos mais comuns é o BSAS [29].

No entanto, o BSAS possui como característica negativa a tendência em gerar um grande número de agrupamentos, inclusive alguns com poucos elementos. Essa tendência pode ser, de certa forma, controlada pelo que é chamado de medida de similaridade. Essa medida é responsável por estabelecer se um modelo de predição será adicionado em um agrupamento já existente, ou se será criado um novo para acomodá-lo. A medida de similaridade é calculada levando em conta os parâmetros estabelecidos para definir a similaridade entre os projetos. O cálculo sempre mantém a referência de quem é o modelo de predição centróide, ou seja, aquele cujas informações seriam a mais central dentre os integrantes daquele agrupamento [29]. O controle sobre o número de agrupamentos que pode ser formado é importante tanto para garantir a existência de recursos computacionais, como para facilitar análise de informações dos agrupamentos. Essa última opção foi a mais relevante para este trabalho e fez com que o número máximo de agrupamento fosse limitado em 20. Com a determinação do limite superior de número de agrupamentos, a medida de similaridade foi calculada a partir do valor de AUC, e fixada em $0,034$.

O algoritmo de agrupamento BSAS foi executado especificamente para cada um dos classificadores escolhidos anteriormente. Na Tabela III, na coluna "Agrupamentos" são listados os agrupamentos obtidos segundo a lista de classificadores escolhidos para serem avaliados neste trabalho. É possível observar que o número de agrupamentos obtidos varia de 14 a 19. Dessa maneira, o número de agrupamentos e os projetos que compõem cada agrupamento variam de acordo com o classificador, conforme ilustrado na Tabela III. Por exemplo, pode-se observar que com o uso do *Naive Bayes* e o BSAS, foram obtidos 19 agrupamentos. Portanto, a partir desses 19 agrupamentos, foi possível construir 19 modelos de predição cruzada de defeitos entre projetos. Na Tabela III pode-se observar que o "Agrup. 01", formado por 6 projetos e 191 instâncias com a utilização de *Naive Bayes*, é diferente do "Agrup. 01", formado por 22 projetos e 991 instâncias com a utilização do *Random Forest*.

Além dos agrupamentos formados pelo BSAS serem diferentes uns dos outros em função do classificador usado, tanto no número de agrupamentos, como nos projetos que compõem o agrupamento (Tabela III), o total de projetos também muda. Conforme pode ser visto na última linha da Tabela III, o número de projetos agrupados foi: para o *Naive Bayes*, 1264; para o *Random Forest*, 1265; para o *Decision Table*, 1267; para o *Simple Logistic*, 1268; e, para o J48, 1269. Isso aconteceu porque o algoritmo BSAS formou agrupamentos que possuem apenas 1 projeto. Quando isso aconteceu, o agrupamento foi descartado.

O segundo produto desta atividade foi a lista de projetos para teste, específicos de cada modelo de predição cruzada. Com a intenção de obter uniformidade nos projetos para teste, foi escolhido para cada modelo de predição cruzada o projeto com o maior número de instâncias. O agrupamento "Agrup. 01", por exemplo, tinha originalmente 7 projetos com o total de 260 instâncias. No entanto, o projeto com o maior número de instâncias chamado `Uclmda`, com 69 instâncias, foi escolhido para ser o projeto de teste desse agrupamento. Assim, resultaram no agrupamento "Agrup. 01", 6 projetos com o total de 191 instâncias.

### *E. Criar modelos de predição cruzada*

O objetivo desta atividade foi desenvolver, treinar e testar os modelos de predição cruzada. Os modelos de predição foram desenvolvidos a partir dos agrupamentos formados. Muito embora cada um dos projetos tenha sido treinado individualmente na atividade anterior, o treino foi novamente efetuado para cada agrupamento com cada um dos classificadores selecionados: 19 modelos treinados com o uso de *Naive Bayes*; 18 modelos treinados com o uso de *Random Forest*; 16 modelos treinados com o uso de *Decision Table*; 15 modelos treinados com o uso de *Simple Logistic*; e 14 modelos treinados com o uso de J48. O teste foi feito com auxílio da lista de projetos para teste obtido na atividade anterior. Os resultados desta atividade foram os modelos de predição cruzada, testados, e as matrizes de confusão dos testes.

Por meio do teste foi possível encontrar o classificador que teve o melhor desempenho, com a observação do valor de AUC obtido para cada modelo de predição cruzada testado com um projeto específico pertencente ao agrupamento que deu origem àquele modelo de predição. Na Tabela IV são apresentados os valores de AUC dos centróides de cada agrupamento juntamente com o teste do projeto selecionado previamente. Os resultados são separados por algoritmo de classificação. Na tabela, valores de AUC $>= 0,7$ são destacados em negrito, e valores de AUC das colunas "AUC Teste" que são maiores do que seus respectivos valores das colunas "AUC Centróide" são apresentados emoldurados.

A fim de averiguar se existe diferença de desempenho entre os classificadores, o teste estatístico de Kruskal-Wallis [25] foi aplicado aos valores AUC obtidos a partir do teste dos projetos de cada um dos modelos de predição cruzada, apresentados nas colunas "AUC Teste" da Tabela IV. O valor obtido para $p$ foi de $0,1541$, maior que $0,05$, que indica que não houve diferença estatística relevante entre as amostras. Com esse resultado, foi necessário usar outra forma para encontrar um classificador com melhor desempenho. A solução empregada foi a contagem da frequência de valores AUC $>= 0,7$. Conforme pode ser visto na Tabela V, que sumariza os resultados mostrados na Tabela IV, 6 modelos de predição cruzada com o uso de *Naive Bayes* (31,58%) obtiveram valor de AUC $>= 0,7$, contra 4 (26,67%) do segundo melhor classificador, o *Simple Logistic*. Portanto, o algoritmo de classificação *Naive Bayes* obteve o melhor resultado de AUC, e foi escolhido para determinar quais modelos de predição cruzada fazem parte da proposta final deste trabalho.

TABELA III
NÚMERO DE PROJETOS E INSTÂNCIAS DE CADA AGRUPAMENTO FORMADO SEGUNDO OS DIFERENTES CLASSIFICADORES.

| Agrupamentos | *Naive Bayes* | | *Random Forest* | | *Decision Table* | | *Simple Logistic* | | J48 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Projetos** | **Instâncias** | **Projetos** | **Instâncias** | **Projetos** | **Instâncias** | **Projetos** | **Instâncias** | **Projetos** | **Instâncias** |
| Agrup. 01 | **6** | **191** | 22 | 991 | **4** | **54** | 9 | 230 | **5** | **84** |
| Agrup. 02 | 120 | 14.998 | 118 | 17.398 | 215 | **28.226** | **225** | **32.943** | **274** | **36.461** |
| Agrup. 03 | 86 | 5.653 | 185 | 17.377 | 109 | 6.321 | 172 | 9.531 | 261 | 22.889 |
| Agrup. 04 | **314** | **42.909** | 68 | 2.657 | 186 | 18.031 | 202 | 28.234 | 144 | 19.430 |
| Agrup. 05 | 100 | 5.725 | 4 | 64 | 26 | 626 | 219 | 24.354 | 71 | 2.692 |
| Agrup. 06 | 5 | 80 | 82 | 10.095 | 45 | 5.639 | 89 | 5.892 | 51 | 4.572 |
| Agrup. 07 | 199 | 29.144 | **234** | **27.247** | **218** | 22.694 | 42 | 5.676 | 78 | 2.928 |
| Agrup. 08 | 153 | 10.957 | 129 | 17.263 | 87 | 4.485 | 159 | 10.240 | 106 | 13.304 |
| Agrup. 09 | 41 | 1.458 | 100 | 6.942 | 34 | 1.460 | 54 | 1.819 | 90 | 8.284 |
| Agrup. 10 | 50 | 2.265 | 48 | 3.003 | 80 | 12.705 | 13 | 308 | 31 | 1.550 |
| Agrup. 11 | 7 | 146 | 24 | 1.926 | 38 | 1.345 | 19 | 530 | 62 | 3.503 |
| Agrup. 12 | 23 | 2.227 | 50 | 2.248 | 109 | 13.005 | 36 | 2.733 | 20 | 501 |
| Agrup. 13 | 75 | 3.986 | 16 | 461 | 51 | 1.818 | 8 | 192 | 30 | 853 |
| Agrup. 14 | 8 | 152 | 103 | 11.430 | 16 | 461 | 19 | 845 | 46 | 2.683 |
| Agrup. 15 | 16 | 467 | 23 | 563 | 6 | 539 | **2** | **74** | – | – |
| Agrup. 16 | 9 | 219 | 54 | 1.784 | 43 | 1.707 | – | – | – | – |
| Agrup. 17 | 17 | 495 | 3 | 124 | – | – | – | – | – | – |
| Agrup. 18 | 15 | 805 | **2** | **17** | – | – | – | – | – | – |
| Agrup. 19 | 20 | 547 | – | – | – | – | – | – | – | – |
| TOTAL | 1.264 | 122.424 | 1.265 | 121.590 | 1.267 | 119.116 | 1268 | 123.601 | 1.269 | 119.734 |

TABELA IV
VALORES DE AUC DOS CENTRÓIDES E DO TESTE DE PROJETO PREVIAMENTE SELECIONADO EM CADA AGRUPAMENTO.

| Agrupamentos | *Naive Bayes* | | *Random Forest* | | *Decision Table* | | *Simple Logistic* | | J48 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **AUC Centróide** | **AUC Teste** | **AUC Centróide** | **AUC Teste** | **AUC Centróide** | **AUC Teste** | **AUC Centróide** | **AUC Teste** | **AUC Centróide** | **AUC Teste** |
| Agrup. 01 | **0,998** | **0,876** | **0,986** | 0,364 | **1,000** | 0,441 | **0,989** | 0,085 | **0,997** | 0,463 |
| Agrup. 02 | **0,824** | **0,729** | **0,752** | 0,583 | 0,667 | 0,500 | **0,784** | **0,822** | 0,666 | 0,567 |
| Agrup. 03 | 0,534 | **0,741** | 0,582 | 0,620 | 0,417 | 0,500 | 0,448 | 0,430 | 0,475 | 0,500 |
| Agrup. 04 | 0,675 | **0,875** | 0,436 | 0,640 | 0,570 | 0,673 | **0,707** | **0,928** | 0,598 | 0,488 |
| Agrup. 05 | 0,490 | 0,563 | 0,140 | 0,560 | 0,102 | **0,975** | 0,632 | **0,718** | 0,061 | 0,492 |
| Agrup. 06 | 0,049 | 0,067 | **0,869** | 0,441 | **0,865** | **0,855** | 0,502 | 0,564 | **0,823** | 0,578 |
| Agrup. 07 | **0,754** | **0,762** | 0,695 | 0,592 | 0,479 | 0,500 | **0,865** | **0,879** | 0,330 | 0,500 |
| Agrup. 08 | 0,587 | 0,551 | **0,801** | 0,646 | 0,243 | 0,500 | 0,548 | 0,615 | **0,757** | 0,430 |
| Agrup. 09 | 0,313 | 0,439 | 0,489 | 0,548 | 0,045 | 0,500 | 0,332 | 0,658 | 0,555 | 0,551 |
| Agrup. 10 | 0,447 | 0,421 | 0,530 | 0,527 | **0,798** | 0,500 | 0,116 | 0,500 | 0,242 | 0,500 |
| Agrup. 11 | 0,156 | 0,618 | **0,924** | **0,808** | 0,153 | 0,500 | 0,000 | 0,000 | 0,396 | 0,537 |
| Agrup. 12 | **0,903** | **0,881** | 0,385 | 0,558 | **0,737** | 0,500 | **0,922** | 0,623 | 0,000 | 0,000 |
| Agrup. 13 | 0,390 | 0,035 | 0,000 | 0,000 | 0,364 | 0,500 | 0,185 | 0,500 | **0,914** | 0,106 |
| Agrup. 14 | 0,080 | 0,605 | 0,635 | 0,473 | 0,000 | 0,000 | 0,253 | 0,527 | 0,167 | 0,500 |
| Agrup. 15 | 0,000 | 0,000 | 0,213 | 0,322 | **0,936** | 0,052 | 0,043 | 0,636 | – | – |
| Agrup. 16 | 0,116 | 0,238 | 0,316 | 0,546 | 0,308 | 0,500 | – | – | – | – |
| Agrup. 17 | 0,259 | 0,615 | **0,948** | **0,806** | – | – | – | – | – | – |
| Agrup. 18 | **0,954** | 0,022 | 0,067 | **0,711** | – | – | – | – | – | – |
| Agrup. 19 | 0,214 | 0,639 | – | – | – | – | – | – | – | – |

O valor que foi adotado de AUC $>= 0,7$ teve como base pesquisas anterior que precisaram estabelecer um parâmetro de comparação para os resultados obtidos com os classificadores [10], [21], [30]. Esse valor é tido como aceitável para considerar que o resultado obtido com um algoritmo de classificação seja de boa qualidade. Em verdade, Malhotra *et al.* [21] menciona que os valores de AUC nos trabalhos de predição de defeitos ficam entre $0,7$ e $0,83$ na maior parte dos casos. Os demais valores representados na Tabela V significam: $<= 0,3$, valores obtidos pelos modelos de predição tidos como de muito baixa qualidade preditiva; entre $> 0,3$ e $<= 0,4$, valores também com baixa qualidade de predição que começam a se aproximar de um modelo randômico; entre $> 0,4$ e $< 0,5$ e entre $>= 0,5$ e $< 0,6$, são valores que se aproximam muito e passam por um modelo randômico; entre $>= 0,6$ e $< 0,7$ são valores que se aproximam de um valor considerado de boa qualidade.

### *F. Avaliar modelos de predição cruzada*

A predição realizada por um modelo necessita ser avaliada e ter seus resultados comparados, para ter seu desempenho identificado [31], [32]. Assim, o objetivo desta atividade foi avaliar os modelos de predição cruzada testados por meio das medidas de avaliação de desempenho calculadas a partir

TABELA V
RESULTADOS DE VALORES AUC DOS MODELOS DE PREDIÇÃO CRUZADA.

| | **<=0,3** | **>0,3 e <=0,4** | **>0,4 e <0,5** | **>=0,5 e <0,6** | **>=0,6 e <0,7** | **>=0,7** | **% >=0,7** |
|---|---|---|---|---|---|---|---|
| *Naive Bayes* | 5 | 0 | 2 | 2 | 4 | 6 | 31,58% |
| *Simple Logistic* | 2 | 0 | 1 | 4 | 4 | 4 | 26,67% |
| *Random Forest* | 1 | 2 | 2 | 7 | 3 | 3 | 16,67% |
| *Decision Table* | 2 | 0 | 1 | 10 | 1 | 2 | 12,50% |
| J48 | 2 | 0 | 4 | 8 | 0 | 0 | 0,00% |

das matrizes de confusão. A avaliação precisa ser feita por indicadores de desempenho que possam aferir o quão bom foi a predição realizada. Segundo Ostrand e Weyuker [31] e Bowees *et al.* [32], as medidas mais utilizadas para este fim são: precisão (do inglês, *Precision*), uma medida de precisão que representa o quanto se está conseguindo classificar corretamente arquivos com defeito, no qual valores próximos de 1 identificam a precisão do modelo; sensibilidade (do inglês, *Recall*), uma medida da proporção de arquivos classificados erradamente como livre de defeito, no qual valores próximos de 1 indicam menor ocorrência de falso negativo; F-Measure, que é a média harmônica entre os valores de precisão e sensibilidade; e, acurácia (do inglês, *accuracy*), uma medida de corretude de como as classes está sendo classificadas corretamente, como verdadeiro positivo ou verdadeiro negativo. Todos esses indicadores podem ser calculados a partir da matriz de confusão, que por si só também pode ser usada para avaliar o desempenho de modelos de predição.

A avaliação do desempenho dos modelos de predição cruzada foi feita com a confrontação do valor de AUC obtido no teste dos projetos selecionados especificamente para os agrupamentos, o que foi chamado de predição local, com o AUC do teste desses mesmos projetos, mas com um modelo de predição cruzada treinado com todos os outros projetos do conjunto de dados, o que foi chamado de predição global. O resultado das execuções dos modelos de predição com o uso do algoritmo de classificação *Naive Bayes*, é exibido na Tabela VI. Nela é possível observar os valores de AUC de três diferentes execuções de modelos: "AUC Centróide", é o valor de AUC com as execuções dos modelos de predição nos projetos que deram origem ao agrupamento, conforme já apresentado na Tabela IV; "AUC Predição Local", é o valor de AUC com a execução dos modelos de predição cruzada obtido com o teste dos projetos, previamente selecionados, dentre os respectivos agrupamentos, a predição local, também apresentado na coluna "AUC Teste" da Tabela IV; e, "AUC Predição Global", é o valor de AUC com a execução dos modelos de predição cruzada testando os projetos previamente selecionados dentro os respectivos agrupamentos contra todos os outros projetos do conjunto de dados, a predição global. Na Tabela VI, valores de AUC $>= 0,7$ são destacados em negrito. Valores de AUC das colunas "AUC Predição Local" que são maiores do que seus respectivos valores das colunas "AUC Centróide" são mostrados em molduras. Seguindo a mesma ideia, valores de AUC das colunas "AUC Predição Global" que são maiores do que seus respectivos valores das colunas "AUC Predição Local" são mostrados em molduras também.

TABELA VI
RESULTADOS DA EXECUÇÃO DOS MODELOS APLICADOS LOCALMENTE NOS AGRUPAMENTOS E GLOBALMENTE.

| **Agrup.** | **Instâncias do Cluster** | **Instâncias Projeto** | **AUC Centróide** | **AUC Predição Local** | **AUC Predição Global** |
|---|---|---|---|---|---|
| Agrup. 01 | 191 | 69 | **0,998** | **0,876** | **0,735** |
| Agrup. 02 | 14998 | 3583 | **0,824** | **0,729** | 0,510 |
| Agrup. 03 | 5653 | 562 | 0,534 | **0,741** | **0,751** |
| Agrup. 04 | 42909 | 2489 | 0,675 | **0,875** | **0,883** |
| Agrup. 05 | 5725 | 224 | 0,490 | 0,563 | 0,575 |
| Agrup. 06 | 80 | 32 | 0,049 | 0,067 | 0,433 |
| Agrup. 07 | 29144 | 2959 | **0,754** | **0,762** | 0,530 |
| Agrup. 08 | 10957 | 628 | 0,587 | 0,551 | 0,555 |
| Agrup. 09 | 1458 | 194 | 0,313 | 0,439 | 0,431 |
| Agrup. 10 | 2265 | 208 | 0,447 | 0,421 | 0,422 |
| Agrup. 11 | 146 | 71 | 0,156 | 0,618 | 0,657 |
| Agrup. 12 | 2227 | 503 | **0,903** | **0,881** | **0,889** |
| Agrup. 13 | 3986 | 491 | 0,390 | 0,035 | **0,835** |
| Agrup. 14 | 152 | 49 | 0,080 | 0,605 | 0,473 |
| Agrup. 15 | 467 | 147 | 0,000 | 0,000 | 0,000 |
| Agrup. 16 | 219 | 68 | 0,116 | 0,238 | 0,220 |
| Agrup. 17 | 495 | 108 | 0,259 | 0,615 | 0,655 |
| Agrup. 18 | 805 | 258 | **0,954** | 0,022 | 0,682 |
| Agrup. 19 | 547 | 97 | 0,214 | 0,639 | 0,567 |

Para verificar os resultados dos modelos de predição cruzada, o teste de Mann-Whitney-Wilcoxon [26] foi aplicado nos valores de AUC mostrados na coluna "AUC Predição Local" e "AUC Predição Global" da Tabela VI. O resultado do teste estatístico foi $p = 0,7592$, maior que $0,05$, mostrou que não há diferença estatística relevante entre os resultados. Dessa forma, optou-se calcular a frequência de valores AUC $>= 0,70$ para identificar os modelos de predição cruzada que obtiveram desempenho adequado. Na Tabela VII são apresentados os resultados desse cálculo, mostrando que o resultado dos testes locais dos modelos de predição obtiveram resultados superiores (31,58%) aos testes efetuados globalmente (26,31%).

## III. DISCUSSÃO DOS RESULTADOS

Existem evidências que mostram que é mais vantajoso o processo de predição cruzada quando ela é feita com projetos agrupados [5], [7]. Em um dos estudos, Menzies *et al.* relata que os resultados com a predição cruzada entre projetos agrupados com alguma similaridade superam os resultados de predições ocorridos em um ambiente sem qualquer tipo de agrupamento. Dessa maneira, esperava-se que a construção dos modelos de predição cruzada com o uso da similaridade de valores AUC, ao invés de por métricas dos projetos, trouxessem alguns benefícios: melhoria da qualidade dos dados do conjunto de dados; eliminação de problemas relacionados ao desbalanceamento de classes da variável dependente (do inglês, *class imbalance*); aumento do poder preditivo dos algoritmos de classificação; e, diminuição do custo de treino dos modelos de predição. A discussão dos resultados

TABELA VII
FREQUÊNCIA DOS VALORES AUC OBTIDOS COM A PREDIÇÃO LOCAL E A PREDIÇÃO GLOBAL.

| | **<=0,3** | **0,3 e <=0,4** | **>0,4 e <0,5** | **>=0,5 e <0,6** | **>=0,6 e <0,7** | **>=0,7** | **% >=0,7** |
|---|---|---|---|---|---|---|---|
| Local | 5 | 0 | 2 | 2 | 4 | 6 | 31,58% |
| Global | 2 | 0 | 4 | 5 | 3 | 5 | 26,31% |

é norteada segundo esses benefícios e dividida em três partes: observações sobre os agrupamentos; observações sobre a QP1; e, observações sobre a QP2.

A respeito dos resultados dos agrupamentos, vale destacar dois pontos interessantes que podem ser vistos pela análise dos resultados da Tabela IV: a dispersão dos valores de AUC dos centróides dos agrupamentos; e, o desempenho dos modelos de predição com o uso de *cross validation* nos projetos dentro dos agrupamentos. Independentemente do classificador usado, os valores de AUC variam muito: *Naive Bayes* varia de $0,000$ a $0,998$; *Random Forest* varia de $0,000$ a $0,986$; *Decision Table* varia de $0,000$ a $1,000$; *Simple Logistic* varia de $0,000$ a $0,989$; e, J48 varia de $0,000$ a $0,997$. O desempenho dos modelos de predição dentro dos agrupamentos em termos de valores de AUC, também independentemente do classificador usado, não é bom: *Naive Bayes* teve 5 valores de AUC $>= 0,7$ nos 19 agrupamentos (26,31%); *Random Forest* teve 6 valores de AUC $>= 0,7$ nos 18 agrupamentos (33,33%); *Decision Table* teve 5 valores de AUC $>= 0,7$ nos 16 agrupamentos (31,25%); *Simple Logistic* teve 5 valores de AUC $>= 0,7$ nos 15 agrupamentos (33,31%); J48 teve 4 valores de AUC $>= 0,7$ nos 14 agrupamentos (28,57%). A média dos valores de AUC dos centróides apresentam valores abaixo de um classificador randômico: *Naive Bayes* obteve $0,460$; *Random Forest* obteve $0,514$; *Decision Table* obteve $0,480$; *Simple Logistic* obteve $0,488$; e, J48 obteve $0,499$. A dispersão dos valores de AUC juntamente com o desempenho instável dos modelos de predição resultou em agrupamentos com número de projetos e instâncias bastante variados, com amplitude também considerável, conforme pode ser visto nos valores em negrito da Tabela III.

Considerando ainda os agrupamentos, ao analisar especificamente o classificador *Naive Bayes*, pode-se perceber que os 5 agrupamentos que obtiveram valores de AUC $>= 0,7$ concentram apenas 28,71% dos projetos do conjunto de dados. Portanto, pode-se concluir que, mesmo com o uso dos dados dos próprios projetos para treino e teste, o desempenho dos modelos de predição não foi satisfatório. Esse fato, juntamente com a informação fato do uso de métodos selecionadores de atributos pode indicar que o conjunto de dados não possui qualidade suficiente para uma boa predição local.

Com relação a resposta para a QP1, que toca no assunto da influência dos classificadores nos resultados dos modelos de predição, a avaliação de desempenho dos modelos feita neste trabalho se assemelha os resultados obtidos por Lessman *et al.* [10]. Com essa avaliação foi possível eleger um classificador vencedor, o *Naive Bayes*, muita embora o teste estatístico conduzido não tenha mostrado diferença relevante entre as amostras. Contudo, vale ressaltar que os resultados dos diferentes classificadores influenciaram a formação dos agrupamentos, tanto na quantidade de projetos como na distribuição desses projetos nos diferentes grupos, conforme mostrado na Tabela III. Alguns algoritmos de classificação mostraram certa tendência a produzir predições randômicas, valores de AUC $> 0,4$ e $< 0,6$, conforme pode ser observado na Tabela V: *Naive Bayes* teve 4 valores de AUC (21,05%); *Random Forest* teve 9 valores de AUC (50,00%); *Decision Table* teve 4 valores de AUC (68,75%); *Simple Logistic* teve 4 valores de AUC (33,33%); e, J48 teve 4 valores de AUC (85,71%). Vale destacar, ainda, que o *Decision Table* teve 10 valores de AUC (62,50%) igual a $0,5$, ou seja, completamente randômicos.

Para complementar a resposta dada à QP2, em que os modelos de predição cruzada propostas obtiveram resultados ligeiramente melhores que o modelo global, alguns pontos mostrados na Tabela VI e na Tabela VII são discutidos aqui. Primeiramente serão discutidos os resultados obtidos pela predição local em comparação com a predição feita com dados dos próprios projetos, representada pelos valores de AUC dos centróides dos agrupamentos. Em um segundo momento, serão confrontados os resultados da predição local e da predição global.

Pode-se observar que a maioria dos valores de AUC da predição local (11 valores, 57,89%) foram maiores do que o valor de AUC do centróide. Algumas das diferenças são sensivelmente maiores, como nos agrupamentos 11, 14, e 19. Assim, os números indicam que o agrupamento de projetos por AUC melhorou o desempenho da predição local com relação ao centróide do agrupamento. Mesmo nos casos em que AUC do centróide foi maior do que o AUC da predição local, pode-se perceber que a predição local obteve sempre resultados $>= 0,7$. Tal fato é uma evidencia de que boas predições dentro do próprio projeto geram agrupamentos capazes de realizar boas predições locais também.

Os casos com resultados menos satisfatórios da predição local contra o centróide podem ser vistos no "Agrup. 15" e no "Agrup. 18". Esses agrupamentos foram fortemente influenciados pelo desbalanceamento de classes. O desbalanceamento de classes traz sérias consequências sobre o poder preditivo dos algoritmos de classificação, e algumas técnicas para enfrentar o problema são propostas na literatura [33]. Como pode ser visto, os valores de AUC das predições relacionadas ao "Agrup. 15" são em todos os casos $0,000$, porque todos os projetos agrupados ali são completamente desbalanceados, por possuírem apenas uma classe de variável dependente (apenas a classe *Buggy*). O "Agrup. 18" também sofre os efeitos dos desbalanceamentos de classes, mas com menor intensidade. O valor do AUC do centróide desse agrupamento teve valor $0,954$ porque a técnica de seleção de atributos (CFS+*Genetic Search*) permitiu ao classificador *Naive Bayes* obter bons resultados durante o processo de *cross-validation*.

Dessa forma, pode-se perceber que o agrupamento pelo

valor de AUC também colaborou para evitar o problema de desbalanceamento de classes, uma vez que ocorreram apenas dois agrupamentos que contem somente 31 projetos compostos de 1272 instâncias (2,444% dos projetos e 1,04% das instâncias do conjunto de dados), muito embora não tenha evitado esse problema por completo.

A comparação dos resultados das predições locais e globais apresentam resultados variados semelhantes. Conforme teste estatístico, não se pode comprovar que existe diferença de desempenho do poder preditivo dos modelos de predição. Assim, apesar da proposta mostrar pontos promissores quando observado o problema do desbalanceamento de classes, do poder preditivo dos modelos locais em relação aos valores de AUC dos centróides e, de certa maneira, na qualidade dos dados, não se pode afirmar que sua aplicação supera o modelo de predição global em poder de predição. Entretanto, alguns outros resultados encontrados no presente trabalho, cabem ser ressaltados. Apesar das diferenças do presente trabalho comparado com o proposto por Zimmermann *et al.* e Turhan *et al.*, apenas como critério de comparação, os resultados obtidos aqui foram significativamente melhores, como é exibido na seção V. Outro ponto importante é que, a aplicação dos modelos de predição cruzada baseados nos agrupamentos de projetos apresenta vantagens quanto ao custo de treino e teste dos modelos. Note-se que, por exemplo, para utilizar os modelos de predição cruzada para predizer um projeto X qualquer, basta apenas treinar o projeto X, conforme feito na atividade "2.3 Construir Modelos de Predição", encontrar o agrupamento adequado em que o projeto X se encaixa e testá-lo com o modelo de predição cruzada referente ao agrupamento já treinado.

## IV. Limitações do trabalho

O presente trabalho apresenta limitações discutidas a seguir. Um primeiro ponto a ser discutido é a qualidade dos dados do conjunto de dados. Conforme discutido, a qualidade dos dados pode influenciar os resultados obtidos pelos classificadores. Neste trabalho, foi assumido que os dados foram minerados e tratados corretamente pelo trabalho de Zhang *et al.* [7]. Além disso, os resultados aqui mostrados são válidos apenas para o conjunto de dados estudado, novos conjuntos de dados devem ser estudados para melhorar a generalização dos modelos de predição cruzada.

Foram escolhidos 5 classificadores para a construção dos modelos de predição aqui propostos. No entanto, faz-se necessário um estudo mais aprofundado, com mais classificadores, conforme feito por Ghotra *et al.* [9], que usou 22 classificadores diferentes. Contrariamente a esse estudo de Ghotra *et al.*, os resultados deste trabalho se assemelham aos de Lessman *et al.* [10], em que não houve diferença estatística relevante entre os modelos de predição cruzada.

De maneira semelhante a escolha dos classificadores, neste trabalho escolheu-se um grupo pequeno de métodos selecionadores de atributos, e apenas um método de agrupamento por similaridade, o BSAS. Faz-se, portanto, necessários mais estudos para entender o impacto dessas técnicas no resultado dos classificadores, no resultado dos agrupamentos e no resultado final dos modelos de predição cruzada.

## V. Trabalhos relacionados

Um ponto importante é que, de maneira geral, é difícil realizar uma comparação direta de resultados entre trabalhos com modelos de predição de defeitos. Essa dificuldade tem diferentes causas, como por exemplo, conjuntos de dados de origens diferentes (dados livres ou dados de projetos proprietários, de certa forma inacessível pela maioria da comunidade científica), diferentes níveis de qualidade dos conjuntos de dados, diferentes organizações dos conjuntos de dados, uso de diferentes classificadores, uso de diferentes maneiras de medir o desempenho do modelo, uso de agrupamento dos projetos ou não, dentre outros.

Nesta seção são discutidas pesquisas relacionadas com o tema predição cruzadas de defeitos em projetos. Muito embora estudos mostrem uma variedade de trabalhos sobre predição cruzada de defeitos, são mostrados nessa seção aqueles considerados mais importantes pelos autores desse trabalho. Foram selecionados os trabalhos de Zimmermann *et al.* [4], Menzies *et al.* [5], Turhan *et al.* [6] e Zhang *et al.* [7], brevemente descritos a seguir.

O trabalho realizado por Zimmermann *et al.* [4] teve como escopo 12 projetos de grande porte. Esses projetos foram agrupados com a formação de 622 pares para execução de predição cruzada entre eles. De todos esses agrupamentos formados, apenas 3,40% desses pares conseguiram predizer defeitos entre si. A forma como Zimmermann *et al.* considera um projeto capacitado para predição de defeito em outros projetos é se seus valores de precisão, sensibilidade e acurácia, obtidos na fase de classificação, são todos acima de 0,75. Apesar do conjunto de dados daquele estudo ser diferente do utilizado por este trabalho, e, apenas como critério de comparação, 42,31% dos 19050 modelos de predição tiveram valores de precisão, sensibilidade e acurácia acima de 0,75. O trabalho de Zimmermann *et al.* apontou que projetos terem o mesmo contexto ou usarem os mesmos processos não é o suficiente para torná-los preditores eficientes. Um exemplo disso foi o resultado da predição entre Internet Explorer e Firefox. Contudo, é importante observar que o próprio autor revela que o contexto pode ter sido afetado pelo subconjunto das métricas escolhidas por especialistas para predição. Esse é um problema não presente neste trabalho, pois a escolha das métricas não foi feita por análise humana e sim pelo processamento dos algoritmos de seleção de atributos.

O estudo conduzido por Menzies *et al.* [5] sugere que o trabalho de predição de defeitos para projetos agrupados por algum tipo de similaridade apresenta melhores resultados do que quando analisados sem qualquer agrupamento. Para comprovar essa afirmação, o autor utilizou um conjunto de projetos disponíveis no repositório PROMISE, sendo eles: Luciane; Xalan; JEdit; Synapse; Tomcat; Velocity e Xerces. A forma de agrupamento foi com uso do algoritmo WHERE, criado pelo autor, que procura agrupar projetos por semelhança entre seus atributos. Essa abordagem de agrupamento entre projetos

foi seguida neste trabalho, contudo, a forma empregada foi diferente com a utilização do valor de AUC como medida de similaridade pelo algoritmo BSAS.

O trabalho de Turhan *et al.* [6] analisou 12 projetos disponíveis no repositório da NASA. Um ponto de atenção sobre esses projetos é que foram desenvolvidos seguindo os rigorosos princípios da ISO-9001 [34], estabelecido pela NASA. Segundo os autores, o fato de seguirem essa norma e o contexto de sua aplicação os diferencia de outros projetos de software livre, o que os torna difíceis preditores para outros projetos. Contudo, mesmo com essa peculiaridade, foi possível evidenciar o benefício do uso de predição cruzada entre projetos para encontrar defeitos. Esse estudo também mostrou um fator de risco interessante que é o aumento da taxa de falso positivos, em uma média de 52%, o que também mostra a dificuldade da transferência de modelos de predição. Apesar do conjunto de dado ser diferente, apenas como ponto de comparação, este trabalho obteve 37,92% de taxa de falsos positivos para os 19050 modelos de predição desenvolvidos. Outra diferença entre os dois trabalhos é que Turhan *et al.* desenvolveu um algoritmo de agrupamento baseado no kNN enquanto este trabalho utilizou o BSAS.

A proposta de Zhang *et al.* [7] é a criação de um modelo universal para predição de defeitos entre projetos. Como já informado na subseção II-B, o presente trabalho utiliza o mesmo conjunto de dados proposto por Zhang *et al.*, com as modificações descritas naquela seção. Para a construção do modelo universal, Zhang *et al.* propõe a transformação dos valores das métricas, presente nos projetos, antes de aplicar a predição cruzada. Dessa forma, os autores avaliam que os projetos estariam com as informações equiparadas o que permitiria o agrupamento de projetos pela similaridade das métricas. Os resultados encontrados sinalizam que, quanto mais informações de contexto disponíveis existem, maiores as chances que o modelo de predição gere bons resultados. Diferentemente do trabalho proposto por Zhang *et al.*, neste trabalho: a transformação dos valores das métricas não é utilizada; e, a medida de similaridade e agrupamento dos projetos também é diferente.

## VI. Trabalhos futuros

Atacar o problema de desbalanceamento do conjunto de dados é um trabalho que traria contribuições significativas para a construção dos modelos de predição. Pode-se encontrar na literatura estudos que visam a melhorar o desempenho dos modelos de predição com desbalanceamento no conjunto de dados, como o uso de combinação de modelos de predição [33], [35]–[39]. No entanto, estudos específicos em modelos de predição cruzada de defeitos não foram encontrados na literatura.

Três outros pontos que se relacionam também podem ser explorados para evoluir este trabalho: algoritmos de classificação; algoritmos selecionadores de atributos; e, algoritmos de agrupamento por similaridade. Conforme citado, estudos são controversos quando se trata do desempenho dos algoritmos de classificação em modelos de predição. A condução de mais estudos, inclusive com experimentos controlados, seria de grande valia. O mesmo pode ser dito a respeito de algoritmos selecionadores de atributos. Estudos controlados sobre o uso dos algoritmos selecionadores de atributos no desempenho dos modelos de predição devem ser conduzidos para aumentar o conhecimento sobre a influência da seleção de atributos nos modelos de predição. Além disso, tais estudos podem ser úteis para melhorar as técnicas de agrupamento por similaridade, uma vez que encontrar projetos similares é uma tarefa bastante importante em modelos de predição cruzada.

## VII. Conclusão

A utilização da predição cruzada de defeito entre projetos é uma alternativa importante a ser considerada durante o desenvolvimento e manutenção para aumentar a percepção de qualidade do software, especialmente para projetos que estão em fase inicial de desenvolvimento, o que permite reduzir a ocorrência de defeitos em seus produtos. Um dos princípios para sua utilização é a identificação de semelhanças entre projetos. Algoritmos de clusterização permitem encontrar padrões de similaridade, sendo uma técnica utilizada noutras pesquisas, que utilizam métricas de software para encontrar projetos semelhantes [4], [7], [8].

Este trabalho propôs uma alternativa ao uso de métricas para estabelecer a similaridade entre projetos que é promissora. Os resultados aqui obtidos indicam que o uso de uma medida de desempenho pode ser vantajoso para o processo de agrupamentos de projetos por similaridade, uma vez que algoritmos de agrupamento podem usar uma diferente variedade de métricas e medida de desempenho para realizar seu trabalho. No entanto, apesar de diminuir o custo de treino e teste durante o processo de predição, os modelos de predição cruzada aqui propostos não obtiveram resultados maiores que um modelo global em termos de desempenho. Além disso, a condução dos estudos sobre o impacto do uso de diferentes classificadores tanto para o processo de agrupamento de projetos por similaridade, quanto para a predição cruzada de defeitos entre projetos, foi um esforço para tentar esclarecer esse assunto que ainda sem consenso na literatura especializada [9], [10]. Portanto, novos estudos se fazem necessários para melhorar os modelos de predição cruzada aqui propostas e também para avaliar o impacto de diferentes classificadores na construção de modelos de predição.

## References

[1] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, "A systematic literature review on fault prediction performance in software engineering," *IEEE Trans. Softw. Eng.*, vol. 38, no. 6, pp. 1276–1304, Nov. 2012. [Online]. Available: http://dx.doi.org/10.1109/TSE.2011.103

[2] M. DÁmbros, M. Lanza, and R. Robbes, "An Extensive Comparison of Bug Prediction Approaches," *Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on*, pp. 31–41, 2010.

[3] C. Catal and B. Diri, "A systematic review of software fault prediction studies," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7346–7354, May 2009.

[4] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: A large scale experiment on data vs. domain vs. process," pp. 91–100, 2009. [Online]. Available: http://doi.acm.org/10.1145/1595696.1595713

[5] T. Menzies, A. Butcher, A. Marcus, T. Zimmermann, and D. Cok, "Local vs. global models for effort estimation and defect prediction," in *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering*, ser. ASE '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 343–351. [Online]. Available: http://dx.doi.org/10.1109/ASE.2011.6100072

[6] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano, "On the relative value of cross-company and within-company data for defect prediction," *Empirical Softw. Engg.*, vol. 14, no. 5, pp. 540–578, Oct. 2009. [Online]. Available: http://dx.doi.org/10.1007/s10664-008-9103-7

[7] F. Zhang, A. Mockus, I. Keivanloo, and Y. Zou, "Towards building a universal defect prediction model," pp. 182–191, 2014. [Online]. Available: http://doi.acm.org/10.1145/2597073.2597078

[8] M. Jureczko and L. Madeyski, "Towards identifying software project clusters with regard to defect prediction," in *Proceedings of the 6th International Conference on Predictive Models in Software Engineering*, ser. PROMISE '10. New York, NY, USA: ACM, 2010, pp. 9:1–9:10. [Online]. Available: http://doi.acm.org/10.1145/1868328.1868342

[9] B. Ghotra, S. McIntosh, and A. E. Hassan, "Revisiting the impact of classification techniques on the performance of defect prediction models," *37th International Conference on Software Engineering (ICSE 2015)*, 2015.

[10] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings," *Software Engineering, IEEE Transactions on*, vol. 34, no. 4, pp. 485–496, July 2008.

[11] K. Faceli, A. C. Lorena, J. Gama, and A. Carvalho, "Inteligência artificial: Uma abordagem de aprendizado de máquina," *Livros Técnicos e Científicos*, p. 381, 2011.

[12] S. Herbold, "Training data selection for cross-project defect prediction," in *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, ser. PROMISE '13. New York, NY, USA: ACM, 2013, pp. 6:1–6:10. [Online]. Available: http://doi.acm.org/10.1145/2499393.2499395

[13] T. L. Graves, A. F. Karr, J. S. Marron, and H. Siy, "Software Change History," *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, vol. 26, no. 7, pp. 653–661, 2000.

[14] T. J. Ostrand, E. J. Weyuker, and R. M. Bell, "Predicting the Location and Number of Faults in Large Software Systems," *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, vol. 31, no. 4, pp. 340–355, 2005.

[15] T. J. McCabe, "A complexity measure," *IEEE Trans. Softw. Eng.*, vol. 2, no. 4, pp. 308–320, Jul. 1976. [Online]. Available: http://dx.doi.org/10.1109/TSE.1976.233837

[16] Y. Jiang, B. Cukic, and T. Menzies, "Fault prediction using early lifecycle data," pp. 237–246, Nov 2007.

[17] A. Bacchelli, M. DÁmbros, and M. Lanza, "Are popular classes more defect prone?" in *Proceedings of the 13th International Conference on Fundamental Approaches to Software Engineering*, ser. FASE'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 59–73. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12029-9_5

[18] M. Pinzger, N. Nagappan, and B. Murphy, "Can developer-module networks predict failures?" pp. 2–12, 2008. [Online]. Available: http://doi.acm.org/10.1145/1453101.1453105

[19] I. S. Wiese, F. R. Côgo, R. Ré, I. Steinmacher, and M. A. Gerosa, "Social metrics included in prediction models on software engineering: a mapping study," in *The 10th International Conference on Predictive Models in Software Engineering, PROMISE '14, Torino, Italy, September 17, 2014*, 2014, pp. 72–81. [Online]. Available: http://doi.acm.org/10.1145/2639490.2639505

[20] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16 – 28, 2014, 40th-year commemorative issue. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0045790613003066

[21] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Applied Soft Computing*, vol. 27, no. 0, pp. 504 – 518, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494614005857

[22] R. Kohavi, "The power of decision tables," pp. 174–189, 1995. [Online]. Available: http://dl.acm.org/citation.cfm?id=645324.649649

[23] P. Refaeilzadeh, L. Tang, and H. Liu, "Cross-validation." pp. 532–538, 2009. [Online]. Available: http://dblp.uni-trier.de/db/reference/db/c.html#RefaeilzadehTL09

[24] T. Fawcett, "An introduction to roc analysis," *Pattern Recogn. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2005.10.010

[25] W. H. Kruskal and W. A. Wallis, "Use of Ranks in One-Criterion Variance Analysis," *Journal of the American Statistical Association*, vol. 47, no. 260, pp. 583–621, 1952. [Online]. Available: http://dx.doi.org/10.2307/2280779

[26] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, 4th ed. Chapman & Hall/CRC, 2007.

[27] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[28] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*, 4th ed. Academic Press, 2008.

[29] J. Kainulainen and J. J. Kainulainen, "Clustering algorithms: Basics and visualization," 2002.

[30] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *Software Engineering, IEEE Transactions on*, vol. 33, no. 1, pp. 2–13, Jan 2007.

[31] T. J. Ostrand and E. J. Weyuker, "How to measure success of fault prediction models," in *Fourth International Workshop on Software Quality Assurance: In Conjunction with the 6th ESEC/FSE Joint Meeting*, ser. SOQUA '07. New York, NY, USA: ACM, 2007, pp. 25–30. [Online]. Available: http://doi.acm.org/10.1145/1295074.1295080

[32] D. Bowes, T. Hall, and D. Gray, "Comparing the performance of fault prediction models which report multiple performance measures: recomputing the confusion matrix." in *PROMISE*, S. Wagner, Ed. ACM, 2012, pp. 109–118. [Online]. Available: http://dblp.uni-trier.de/db/conf/promise/promise2012.html#BowesHG12

[33] T. M. Khoshgoftaar, N. Seliya, and D. J. Drown, "Evolutionary data analysis for the class imbalance problem," *Intell. Data Anal.*, vol. 14, no. 1, pp. 69–88, Jan. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1735163.1735168

[34] D. Ince, *ISO 9001 and software quality assurance*. McGraw-Hill, Inc., 1994.

[35] P. A. Flach and S. Wu, "Repairing concavities in roc curves," in *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, ser. IJCAI'05. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005, pp. 702–707. [Online]. Available: http://dl.acm.org/citation.cfm?id=1642293.1642406

[36] M. Majnik and Z. Bosnić, "Roc analysis of classifiers in machine learning: A survey," *Intell. Data Anal.*, vol. 17, no. 3, pp. 531–558, May 2013. [Online]. Available: http://dx.doi.org/10.3233/IDA-130592

[37] P. Wang, K. Tang, T. Weise, E. P. K. Tsang, and X. Yao, "Multiobjective genetic programming for maximizing roc performance," *Neurocomput.*, vol. 125, pp. 102–118, Feb. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2012.06.054

[38] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, no. 1-2, pp. 1–39, Feb. 2010. [Online]. Available: http://dx.doi.org/10.1007/s10462-009-9124-7

[39] R. Fandos, C. Debes, and A. M. Zoubir, "Resampling methods for quality assessment of classifier performance and optimal number of features," *Signal Process.*, vol. 93, no. 11, pp. 2956–2968, Nov. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.sigpro.2013.05.004