A Importância Dos Desenvolvedores De Software Sob A Perspectiva Dos Supervisores

Guilherme Costantin Tângari

Universidade Federal de Uberlândia
Faculdade de Computação
Programa de Pós-Graduação em Ciência da Computação

Uberlândia
2014

Guilherme Costantin Tângari

A Importância Dos Desenvolvedores De Software Sob A Perspectiva Dos Supervisores

Dissertação de mestrado apresentada ao Programa de Pós-graduação da Faculdade de Computação da Universidade Federal de Uberlândia como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Ciência da Computação

Orientador: Nome completo do orientador
Coorientador: Nome completo do coorientador

Uberlândia
2014

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU, MG, Brasil.

A474m Sobrenome, Nome do aluno, 1979-

2014 Título do Trabalho / Nome e Sobrenome do aluno. - 2014.
81 f. : il.

Orientador: Nome do Orientador.

Dissertação (mestrado) - Universidade Federal de Uberlândia, Programa de Pós-Graduação em Ciência da Computação.
Inclui bibliografia.

1.Computação - Teses. 2. Simulação (Computadores) - Teses. I. Sobrenome, Nome do orientador. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDU: 681.3

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Os abaixo assinados, por meio deste, certificam que leram e recomendam para a Faculdade de Computação a aceitação da dissertação intitulada "**Título do trabalho**" por **Nome do aluno** como parte dos requisitos exigidos para a obtenção do título de **Mestre em Ciência da Computação.**

Uberlândia, ___ de _____ de _____

Orientador: _____
Prof. Dr. Nome do orientador
Universidade Federal de Uberlândia

Coorientador: _____
Prof. Dr. Nome do coorientador
Universidade Federal de Uberlândia
(quando houver)

Banca Examinadora:

_____
Prof. Dr. Membro da banca 1
Instituição de Ensino Superior

_____
Prof. Dr. Membro da banca 2
Instituição de Ensino Superior

À minha família e à minha futura esposa.

Agradecimentos

Primeiramente gostaria de agradecer aos meus pais, Gilberto e Valéria, que fizeram uma enorme esforço para que eu pudesse ter acesso à educação que tive. Saibam que serei eternamente grato por todos os sacrifícios que fizeram e nenhuma herança no mundo supera a base educacional que vocês me proporcionaram. Sem dúvida nenhuma, vocês são os grandes responsáveis por eu ter chegado até onde cheguei e divido todos os méritos com vocês. Agradeço também à minha irmã Larissa por todo o apoio nesses últimos anos, também fundamental para que eu pudesse seguir em frente. De maneira especial, também gostaria de agradecer à minha noiva, que me deu todo o suporte, mental e estrutural, e me deu forças quando elas faltaram para seguir em frente. Liana, com certeza sem você esse caminho teria sido muito mais árduo, talvez impossível de ser traçado. Com seu apoio e sua motivação, consegui chegar ao fim dessa jornada, e também divido os méritos com você. Também gostaria de deixar um abraço especial para toda sua família, que sempre me deram apoio incondicional. Gostaria de agradecer também à empresa na qual trabalho atualmente, o MahaGestão, que apostou em mim como profissional e forneceu a flexibilidade necessária no decorrer do curso viabilizando assim esse mestrado. Não menos importante, gostaria de agradecer a todos os meus amigos, aos de longa data, que me acompanham desde o ensino básico e que continuam firme ao meu lado onde sempre estiveram sempre que precisei, às amizades feitas ao longo da minha graduação, que também já se tornaram amizades de longa data, e que compartilham de uma realidade de estudos e trabalhos semelhantes, sempre apoiando uns aos outros, e também à todos os outros grupos de amigos, do Maha, dos casais (amiguinhos), da Charanga, da minha noiva que eu adotei como meus (chicas), todos vocês foram muito importantes nessa jornada. Um salve especial ao Lucas (Lucão) pela sua colaboração com meu trabalho. E por último, gostaria de agradecer pela orientação do professor Marcelo Maia, seu apoio e compreensão foi fundamental para a conclusão desse trabalho e do mestrado como um todo.

"Sua vida pode ser dividida em dois períodos: antes de agora e a partir de agora."
(Prof. Obvious Stating)

Resumo

Várias empresas de tecnologia usam a quantidade de entregas como métrica de avaliação de performance de desenvolvedores de software. Esse é o conceito clássico de produtividade, e ainda é amplamente usado pelas empresas hoje em dia. Também é bastante comum misturar o conceito de importância com produtividade. Porém, a importância de um desenvolvedor para a empresa e, mais especificamente, o time em que trabalha não está apenas relacionado com a quantidade de linhas de código produzidos. Existe uma variedade de fatores que contribuem para a relevância de um desenvolvedor dentro de uma organização. Esse trabalho visa mapear esses fatores, medir quais dentre esse fatores possuem maior influência nas empresas atualmente e propor um modelo de avaliação da importância dos desenvolvedores que considere mais do que apenas as entregas. Levantamos, baseados em estudos passados, dezesseis fatores que mais tendem a participar da avaliação de importância dos desenvolvedores. Descobrimos que, dentre esses fatores, alguns são mais importantes que os outros no momento da avaliação, bem como uma variação nos fatores mais importantes quando olhamos sob a ótica de uma determinada empresa ou time. Nós também construímos um classificador de alta acurácia que pode indicar a importância do desenvolvedor baseado em uma série de atributos.

Palavras-chave: Importância do desenvolvedor. reconhecimento de padrões. produtividade. fatores humanos..

Abstract

Several technology companies use the amount of deliveries as evaluation metric for the software developer's performance. This is the classical concept of productivity, and still is widely used by the companies nowadays. It is also quite common to confuse the concepts of importance and productivity. But the importance of the developer for the company, and more specifically, for the team he works, isn't related only with the amount of line of codes produced. There is a variety of factors that contribute to the relevance of a developer inside an organization. This work aims to map those factors, measure which ones has greater influence in today's companies and propose an evaluation model of the developer's importance that considers more than just deliveries. We raised, based on past studies, sixteen factors that are more likely to be used in the developer's importance evaluation. We figured out that, between those factors, some are more important than others at the evaluation moment, and that there is a variation in the most important factors when we analyze under the perspective from different companies or teams. We also built a high accuracy classifier that can measure the developer's importance based on a series of factors.

Keywords: Developer's importance. pattern recognition. productivity. human factors..

Lista de ilustrações

Lista de tabelas

Lista de siglas

Sumário

Capítulo 1

Introdução

1.1 Considerações Iniciais

Atualmente, as empresas em geral estão investindo em técnicas para aumento de produtividade, com o intuito de aumentar a competitividade no mercado, e isso não é diferente para a indústria de software, que permanece investindo em métodos, ferramentas e melhores práticas para ter um ganho na produção de seu software (de Barros Sampaio et al., 2010). Porém, diferentemente do hardware, que tem ganhos em ordens de magnitude de preço e performance por década, a produção de software parece ter dificuldade em evoluir (BOEHM, 1987). As taxas de produtividade atuais são similares às taxas de décadas atrás (uma ou duas linhas de código por homem-hora)(BOEHM, 1987). Brooks et al. (Brooks Jr, 1987) afirma ainda que não há nenhuma técnica, seja de tecnologia ou de gerenciamento, que por si só prometa um aumento de uma ordem de magnitude na produtividade, simplicidade e confiabilidade do software.

Os métodos tradicionais para se medir produtividade em desenvolvimento de software são baseados em linhas de código (LOC) e pontos de função (FP)(WAGNER; RUHE, 2008), por exemplo, a quantidade de LOC ou FP produzidos por um desenvolvedor em uma hora. Uma definição um pouco mais abstrata coloca produtividade como sendo os outputs entregues pelos inputs consumidos, podendo os outputs ser LOC, FP ou alguma outra saída considerada relevante, e inputs como recursos utilizados para produzir aquela saída (tempo, pessoal, etc), como mostrado na Equação 1 (BOEHM, 1987; WALSTON; FELIX, 1977; YU; SMITH; HUANG, 1991).

$$\text{Produtividade} = \frac{\text{outputs produzidos pelo processo}}{\text{inputs consumidos pelo processo}} \quad (1)$$

1.2 Problema e Motivação

Dentro das empresas de TI, geralmente é encontrado um departamento de recursos humanos ou desenvolvimento pessoal. Esses departamentos são responsáveis pela devida

remuneração e determinação de cargos e posições dentro das empresa, e geralmente o fazem com base em avaliações de desempenho vinda dos supervisores responsáveis de cada área. Geralmente, encontramos um ou mais supervisores ou gerentes responsáveis na área de desenvolvimento, comumente conhecida pelos nomes de indústria ou fábrica de software.

Utilizar apenas o método tradicional citado na seção 1.1 para avaliar o desempenho de um desenvolvedor pode ser muito negativo para a empresa. Esse método de avaliar apenas LOC por exemplo é considerado primitivo e produz resultados que não estão de acordo com a realidade (SYMONS, 2010). Quantidade de linhas de código não leva em consideração o esforço necessário para escreve-las, nem o conhecimento que serviu de base para o mesmo acontecer. Problemas complexos, por exemplo, geralmente necessitam de um desenvolvedor mais experiente para serem resolvidos, e muitas vezes, não são necessários muitas linhas de código para o fazer.

Existem outras noções de produtividade que também não são levadas em consideração ao avaliar apenas linhas de código, por exemplo, um desenvolvedor com conhecimento em uma ferramenta muito específica, ou o supervisor, podem ser consultados frequentemente por outros desenvolvedores, agilizando e melhorando o processo de desenvolvimento de outros desenvolvedores, logo, esse desenvolvedor/supervisor possui uma produtividade indireta, que não é mensurada no método tradicional pois quando se está ajudando e ensinando, não se está escrevendo linhas de código.

Características comportamentais , encontradas no perfil do desenvolvedor, também podem influenciar na avaliação do desenvolvedor, aumentando sua relevância para a empresa. Pró-atividade, criatividade e liderança são alguns exemplos de características que geralmente fazem diferença na carreira de um funcionário dentro de uma organização. O método clássico também falha em não considerar esses quesitos, pois a avaliação de linhas de código não demonstram inovação, alinhamento com o negócio, etc.

Vários estudos se dedicam a levantar fatores que influenciam na produtividade, tanto no desenvolvimento quanto na manutenção de software, (de Barros Sampaio et al., 2010; WAGNER; RUHE, 2008; CALOW, 1991; VOSBURGH et al., 1984), dentre outros. Entender esses fatores e ter um mecanismo de avaliação de produtividade justo é muito importante para as empresas que desenvolvem qualquer tipo de software.

Retenção de talentos (BOEHM et al., 2000; CHATZOGLOU; MACAULAY, 1997; DEMARCO; TIMOTHY, 1987; GUZZO, 1988; SCUDDER; KUCIC, 1991; WOHLIN; AHLGREN, 1995; WOHLIN; ANDREWS, 2001) e motivação da equipe (BOEHM, 1987; DEMARCO; TIMOTHY, 1987; BOEHM, 1984; JONES, 2000; SHARP et al., 2009; BOEHM et al., 1982; BOEHM; PAPACCIO, 1988; HANTOS; GISBERT, 2000), por exemplo, são duas questões de extrema relevância para qualquer time. Motivação é tido como um dos fatores chave para o sucesso do software (SHARP et al., 2009), e software é feito por pessoas, e pessoas, quando tem seu trabalho reconhecido e valorizado tendem a produzir

mais e melhor. Uma avaliação de desempenho que considere apenas um aspecto, como a quantidade de entregas, e não leva em conta a dificuldade e a finalidade do código, e o relacionamento do dia a dia com os colegas e com a empresa, não é uma avaliação justa, que faz com que ocorra a desmotivação individual ou geral da equipe. Rotatividade de funcionários é um problema comum à empresas de software(ABDEL-HAMID; MADNICK, 1991; WALLACE; KEIL; RAI, 2004), e uma alta taxa de rotatividade pode levar à uma consequente diminuição de produtividade devida a uma perda de conhecimento (MELO et al., 2011; CORAM; BOHNER, 2005), além do aumento de custo com contratação e treinamento e o mais importante, a perda de talentos que vão em busca de reconhecimento em outro empresa (até mesmo no concorrente).

Várias empresas estão começando a ganhar consciência dessas questões e estão motivadas a melhorar a forma que são feitas as avaliações de desempenho dos desenvolvedores. Esse trabalho visa investigar como os supervisores entendem a noção de importância, indicando quais fatores são mais relevantes em suas avaliações sobre os desenvolvedores. Estamos procurando responder essas perguntas:

1. Quais são os critérios mais importantes usados pelos supervisores em sua classificação sobre os desenvolvedores?
2. É possível construir um classificador de desenvolvedores de alta acurácia, usando os critérios propostos? Essa pergunta pode ser refinada em duas novas perguntas:
   a) É possível se obter um classificador genérico, i.e., independente de empresa?
   b) É possível se obter classificadores customizados para cada empresa?

1.3 Objetivos e Contribuição

Como mencionado na seção 1.1, o desempenho de um desenvolvedor é muito relacionado com sua produtividade, que possui um conceito clássico de quantidade de entregas. Porém, os supervisores e gerentes que participam ativamente da sua área de desenvolvimento e convivem com os desenvolvedores que ali trabalham, possuem percepções diferentes sobre cada membro de seu time. Isso nada mais é que uma avaliação subjetiva de desempenho, que não considera apenas a produção individual resultante, mas diversas outras características que fazem que o desenvolvedor possua uma determinada importância na visão do seu supervisor.

Entendemos que a métrica mais comum hoje é a "produtividade" no sentido de quantidade de entregas. Porém, sabemos que muito mais é levado em consideração ao avaliar a importância de um determinado desenvolvedor, o problema é que essa avaliação geralmente acontece de uma forma subjetiva por parte do supervisor, isto é, cada supervisor, de acordo com a vivência, identifica quais os desenvolvedores mais importantes devido à sua percepção, sem a orientação de métricas bem estabelecidas.

Fizemos um levantamento, baseado em estudos passados, de várias métricas que podem influenciar na avaliação de cada desenvolvedor. Realizaremos uma pesquisa com sujeitos humanos (supervisores de áreas de desenvolvimento representando empresas ou times), onde cada um desses sujeitos humanos irá avaliar individualmente os desenvolvedores seguindo as métricas levantadas. Iremos então analisar o conjunto de dados resultante da aplicação da pesquisa com intuito de identificar um padrão nas avaliações dos desenvolvedores. Dessa forma, pretendemos apontar quais as métricas que mais influenciam na avaliação dos supervisores sobre os desenvolvedores, e também obter um classificador de importância dos desenvolvedores, utilizando essas métricas mais relevantes, que esperamos seja de alta acurácia. Os supervisores que preencherem uma quantidade significativa de dados (10 ou mais questionários) também terão sua empresa/time analisados individualmente, tanto na descoberta das métricas mais relevantes como na construção do classificador de importância do desenvolvedor.

1.4 Estrutura da Dissertação

Além do presente capítulo introdutório, esta pesquisa apresenta-se desenvolvida e documentada dentro da seguinte estrutura organizacional:

Capítulo 2: Referencial Teórico

Nesse capítulo apresentaremos os conceitos importantes que serão abordados ao longo desse trabalho, bem como outros estudos que atuaram em cima desse mesmo tópico e que serviram de base para produção desse estudo.

Capítulo 3: Metodologia

Nesse capítulo será apresentado a metodologia usada para conduzir a pesquisa com sujeito humanos, mostrando a criação do conjunto de critérios para os supervisores avaliarem os desenvolvedores e a aplicação da pesquisa. Será mostrado também a estratégia de seleção de características utilizada para identificar quais critérios possuem maior relevância no momento da avaliação do supervisor, e a estratégia para a construção do classificador de importância dos desenvolvedores.

Capítulo 4: Resultados Gerais (todas empresas)

Nesse capítulo apresentaremos o resultado da aplicação da pesquisa, mostrando o número de supervisores e empresas envolvidos na pesquisa e a quantidade de avaliações de desenvolvedores obtidas. Mostraremos também o resultado da aplicação dos algoritmos de seleção de características e de classificação para todas as empresas.

Capítulo 5: Resultados individuais (por empresa)

Nesse capítulo apresentaremos o resultado da aplicação da pesquisa, aplicação dos

algoritmos de seleção de características e de classificação para as empresas que atingirem o número mínimo de avaliações de desenvolvedores.

Capítulo 6: Discussão

Aqui discutiremos os resultados obtidos da aplicação dos algoritmos de seleção de características e classificação, apresentando também algumas ameaças à validade desse estudo.

Capítulo 7: Conclusões

Por fim, baseada nos resultados obtidos, apresentaremos a conclusão do trabalho e levantaremos algumas possibilidades de trabalhos futuros que possam ter como base esse estudo.

Capítulo 2

Referencial Teórico

2.1 Considerações Iniciais

O nosso estudo gira em torno do conceito de importância dos desenvolvedores, sob as perspectivas dos seus supervisores. Porém, notamos que esse conceito se mistura com conceitos de produtividade, performance ou eficiência. Vários estudos, que serviram de base para levantarmos as métricas para os supervisores avaliarem os desenvolvedores, eram na verdade estudos sobre produtividade e controle de custos e qualidade de software.

Ao longo do tempo, vários estudos se dedicaram a encontrar e classificar os chamados fatores de produtividade (VOSBURGH et al., 1984; WALSTON; FELIX, 1977; BROOKS, 1981; HANSON; ROSINSKI, 1985; JONES, 1986; BOEHM; PAPACCIO, 1988; JONES, 1997; SCUDDER; KUCIC, 1991; BANKER; DATAR; KEMERER, 1991; BOEHM, 1984; BANKER; DATAR; KEMERER, 1987; SCACCHI; HURLEY, 1995; BRIAND; EMAM; BOMARIUS, 1998; JONES, 2000; LOKAN, 2001; CLINCY, 2003; WAGNER; RUHE, 2008; de Barros Sampaio et al., 2010). Em COCOMO (BOEHM et al., 2000) (Constructive Cost Model) considerado um dos estudos mais proeminentes na área, Boehm classifica os fatores de produtividade diversas categorias (produto, projeto, etc.). Abstraindo ainda mais as categorias, ele classifica os fatores em fatores técnicos e fatores do tipo "soft" (WAGNER; RUHE, 2008), que são os fatores não-técnicos que influenciam na produtividade.

De Marco e Lister (DEMARCO; TIMOTHY, 1987) apontaram que "talvez os maiores problemas de trabalhar com sistemas não são tanto tecnológicos quanto sociológicos". Em seu estudo, por exemplo, eles afirmam que um dos fatores que mais influencia na produtividade é a rotatividade de funcionários (como ressaltado na seção 1.2). Em suma, eles proporcionaram o primeiro e mais compreensível estudo sobre os fatores "soft" influenciando na produtividade dos desenvolvedores de software (WAGNER; RUHE, 2008).

Como nosso trabalha se dedica a medir a importância do desenvolvedor, focaremos mais nos fatores do tipo "soft" do que nos fatores do tipo técnico (que envolvem fatores relativos à produto e projeto, como tamanho do software, levantamentos de requisitos ,

etc (de Barros Sampaio et al., 2010)).

2.2 Levantamento de Características

Nessa seção mostraremos então quais os fatores do tipo "soft" utilizaremos para traduzir do abstrato para o concreto a avaliação dos supervisores sobre seu desenvolvedores. Mostraremos também estudos onde esses fatores foram referenciados, sob o contexto de avaliação de fatores que influenciam na produtividade dos desenvolvedores de software. Esses fatores, como mostrado na seção 3.2.1, serão utilizados para compor as métricas do nosso modelo Goal-Question-Metric (GQM), que nos auxiliará na condução da nossa pesquisa com as empresas.

Primeiramente, é importante citar que agrupamos os fatores levantados por semelhança de conceito de forma a deixar o levantamento mais conciso. Esses grupos ajudarão na elaboração das perguntas do modelo GQM. São eles:

Características técnicas

Aqui agrupamos os fatores relativos às habilidades do desenvolvedor. Dentre os fatores mais citados na literatura, selecionamos quatro, que contemplam a experiência do desenvolvedor em um determinado método ou ferramenta, se ele possui conhecimento especializado em uma determinada tecnologia, ou mesmo se possui uma diversidade de conhecimentos em variadas tecnologias, e sua capacidade em resolver problemas complexos.

Características comportamentais (quando em equipe)

Encontramos diversos estudos que mostram como a coesão e a comunicação de um time de desenvolvimento influencia na produtividade dos desenvolvedores pertencentes à esse time. Dividimos as métricas então sobre o comportamento do desenvolvedor quando encontra um problema (se ele é do tipo introspectivo que tenta resolver sozinho ou comunicativo que logo consulta alguém do time), quando algum colega solicita ajuda e de um modo geral como é a sua comunicação com seus colegas de time.

Características Individuais (encontradas no perfil do desenvolvedor)

Aqui entramos um pouco na ciência de Gestão de Pessoas. Para usar como fatores de importância do desenvolvedor, mapeamos algumas das competências citadas como mais importantes para as organizações hoje em dia. Competência, segundo Chiavenato (CHIAVENATO, 2008), constitui um repertório de comportamentos capazes de integrar, mobilizar, transferir conhecimentos, habilidades, julgamentos e atitudes que agregam valor econômico à organização. As competências selecionadas são mostradas na Tabela 1 e os estudos que as referenciam geralmente estão ligados a estudos de gestão por competências.

Compromisso com o Time/Empresa

Amplamente encontrado em literatura relativa à metodologias de desenvolvimento ágil, por serem fatores que baseiam a filosofia ágil que visa melhorar a produtividade de um time, estão os fatores mapeados à essa categoria. Foco nos clientes, nos resultados e organização são alguns deles. Outro fator também se encaixa bem à essa categoria, que é o tempo de trabalho do desenvolvedor na empresa, que representa a fidelização do funcionário.

Tabela 1 – Levantamento dos fatores de importância por grupo de semelhança

Agrupamentos	**Fatores de importância**	**referência dos estudos onde os fatores são citados, mostrados na Tabela 2**
Características técnicas	Experiência relevante	(1)
	Conhecimento especializado	
	Diversidade de habilidades	
	Capacidade de resolução de problemas complexos	
Características comportamentais	Principal comportamento do desenvolvedor ao encontrar um problema	(2)
	Disposição para ajudar colegas quando solicitado	
	Comunicação com os colegas	
Características individuais	Liderança	(3)
	Pró-atividade	
	Criatividade	
	Empreendedorismo	
Compromisso com o Time/Empresa	Foco no cliente	(4)
	Foco nos resultados	
	Organização e planejamento	
	Tempo de trabalho	

Tabela 2 – Revisão da literatura correlata aos fatores de importância

Referência dos fatores	**Estudos onde os fatores são citados**
1	(CHATZOGLOU; MACAULAY, 1997; COLE, 1995; JONES, 1986; MAXW
2	(ALPER; TJOSVOLD; LAW, 2000; BOEHM et al., 2000; CHATZOGLOU;
3	(CHIAVENATO, 2008; LALSING; KISHNAH; PUDARUTH, 2012; Faria Su
4	(LALSING; KISHNAH; PUDARUTH, 2012; MELO et al., 2011; Faria Sueli

2.3 Data Mining e Classificação

Segundo as palavras do Dr. Ian H. Witten (HOLMES; DONKIN; WITTEN,), Mineração de dados se trata de pegar os dados brutos e transformá-los em algo mais útil, como informação ou predições de algo que pode vir a acontecer, que podem ser úteis no mundo real.

Dr. Ian é professor do departamento de Ciência da computação da Universidade de Waikato, que faz várias pesquisas sobre aprendizado de máquinas, e autor de trabalhos importantes na área (WITTEN; FRANK, 2005). Como resultado dessas pesquisas, foi criada uma ferramenta chama WEKA (Waikato Environment for Knowledge Analysis)(HOLMES; DONKIN; WITTEN,) que será usada para gerar os resultados vistos nessa dissertação. Weka é um software open-source que contém um grande número de algoritmos para classificação, pré-processamento de dados, seleção de características, clusterização, regras de associação, etc. Faremos uso desses algoritmos de aprendizado de máquina para realizar a mineração de dados no conjunto de dados resultante da nossa pesquisa.

O uso de algoritmos de aprendizado de máquinas não é novo no estudo sobre a produtividade dos desenvolvedores. Sharpe et al. (SHARPE; CANGUSSU, 2005) utilizou de algoritmos de reconhecimento de padrões para classificador os desenvolvedores de acordo com sua produtividade, mas especificamente, utilizou algoritmos de clusterização. No nosso caso, como iremos obter a avaliação dos desenvolvedores pelos supervisores em cima de um conjunto pré-definido de classes, utilizaremos algoritmos de classificação.

Capítulo 3

Metodologia

3.1 Considerações Iniciais

Para investigar a prática atual de avaliação de desenvolvedores, nós precisamos olhar mais profundamente em como as empresas o fazem. No propósito de conduzirmos esse estudo, decidimos executar uma pesquisa com sujeitos humanos para extrair a informação desejada e analisá-la. Nesse estudo nós pedimos para os respondentes para primeiro classificar o desenvolvedor (em níveis de importância), e depois preencher o resto do formulário com as características do desenvolvedor.

Para analisar os dados obtidos, nós decidimos em usar métodos de classificação para obtermos uma visão clara em como as características afetam a classificação do supervisores, e como produto, nós talvez consigamos obter um classificador que pode ser usado para ajudar os supervisores a ganhar insights de como melhorar a produtividade geral do time.

Esta seção se dedica a explicar o desenho desse experimento, mostrando a criação e a aplicação da pesquisa com os supervisores, e mostrar também como conduzimos a análise dos dados obtidos dessa pesquisa, incluindo a análise de critérios e a construção do classificador.

3.2 Pesquisa

3.2.1 Goal-Question-Metric

Nós usamos uma abordagem chamada Goal-Question-Metric(GQM)[53] que nos ajudou a compor nossa pesquisa. GQM é uma abordagem do tipo top-down, que é baseada no pressuposto que primeiro, para se medir algo, você precisa especificar objetivos (goal), dos quais é possível derivar perguntas (question), e então, especificar as métricas (metric) que precisam ser coletadas para responder essas perguntas.

Para atingir o propósito de um objetivo, nós temos que determinar 3 coordenadas:

a) **Questão *(issue)*:** Sobre qual questão/assunto você está lidando;

b) **Objeto/Processo *(object/process)*:** Qual é o objeto central da análise;

c) **Ponto de vista *(ViewPoint)*:** Sob a perspectiva de quem a análise está sendo feita.

A Tabela 3 mostra nosso modelo GQM, com o nosso propósito, as perguntas derivadas do mesmo e as métricas levantadas para responder essas perguntas.

3.2.2 Aplicação da Pesquisa

Nós aplicamos a pesquisa de maneira remota, para dar a liberdade necessárias para que os respondentes respondessem as perguntas. Para isso, nós usamos a ferramenta de formulários do Google (Google Form).

Para preservar a privacidade das empresas participantes, nós não solicitamos nenhum tipo de identificação tanto para o respondente como do desenvolvedor sendo analisado. A única identificação que solicitamos foi o nome a empresa de onde vinham as avaliações. Foi necessário solicitar esse dado para uma investigação mais profunda nos casos em que as empresa atingiam o mínimo de 10 desenvolvedores avaliados.

3.2.3 Caracterização dos respondentes

A pesquisa foi projetada para ser aplicada em empresas que possuem uma área de desenvolvimento de software, com uma estrutura hierárquica mínima onde existisse o cargo de supervisor, ou gerente, ou engenheiros-chefe, etc (para futuras referências, nós chamaremos essa pessoa de supervisor).

Os respondentes da pesquisa são então supervisores de times de desenvolvimento. Nós entendemos que eles são as pessoas certas para o fazer porque, diferente do dono da empresa, eles estão perto o suficiente do dia a dia de trabalho, e conseguem julgar quais os desenvolvedores mais importantes e porque, mesmo que eles não use um método formalizado para tal. Eles devem responder uma avaliação para cada desenvolvedor, i.e., se eles estão avaliando 10 desenvolvedores para atingir o mínimo para obter uma avaliação individual da empresa, eles precisam responder 10 questionários.

3.3 Seleção de Características

No intuito de conduzir a análise dos critérios para determinar quais fatores são os mais relevantes e possuem maior influência na avaliação do supervisor, nós usamos a ferramenta WEKA[50].

Vários problemas reais, como o nosso, possuem diversas características envolvidas (mostradas na Tabela 3), e apenas algumas dentre elas são relevantes para o objetivo fi-

nal [54], no nosso caso, classificar a importância de um desenvolvedor. Para resolver essa problema, nós iremos usar uma estratégia chamada seleção de características (do inglês, Feature Selection), onde nós iremos selecionar um subconjunto de características para focar nossa atenção, e ignorar o restante para acelerar o processo de aprendizado, aprimorar a qualidade do classificador e atingir a melhor acurácia do algoritmo de aprendizado de máquinas [54], [55].

O algoritmo que iremos usar é chamado GainRatioAttributeEval, que é um avaliar de atributo único, que avalia os atributos um a um independentemente e os ordena de acordo com sua influência sobre a classe. Nossa seleção de características vai então escolher, baseado nessa ordenação, e isso nos permite eliminar atributos irrelevantes. Esse método, como não avalia um subconjunto de atributos, não elimina atributos redundantes (apenas os irrelevantes), mas isso não é um problema para porque nós conhecemos todos os atributos e esse tipo de avaliador não necessita de um método de buscam o que o torna muito rápido em sua execução.

3.4 Classificação

Como resultado de nossa pesquisa, nós teremos um conjunto com várias avaliações de supervisores sobre os desenvolvedores. Nesse conjunto nós aplicaremos algoritmos de aprendizado de máquinas, no intuito de gerar um classificador de importância. Os algoritmos de aprendizado de máquinas precisam de dois conjunto de dados: um para realizar o treinamento e um para testar o resultado, para verificar a acurácia do classificador. A Figura 1 mostra o processo que melhor representa esse cenário.

Existe uma variedade de métodos para avaliar a performance de um classificador. Nós podemos, por exemplo, selecionar uma porcentagem de um conjunto de dados para aplicar os algoritmos de aprendizado de máquinas e usar o resto para teste. O método de avaliação padrão e mais confiável é chamado cross-validation. No nosso caso, nós usamos o 10-fold cross-validation que divide o conjunto em 10 partes iguais (as chamadas folds), usa 9 partes dessa divisão para treinamento um guarda uma parte para teste, e depois, faz isso mais 9 vezes, sempre alternando o pedaço utilizado para teste (como mostrado na Figura 2), assim, uma determinada parte é usada 9 vezes para treinamento e uma para teste. O resultado será a média das 10 vezes que o algoritmo executou.

Os algoritmos de aprendizado de máquina que nós iremos usar serão o J48, um classificador em árvore, e o NaïveBayes, uma classificado bayesiano.

J48 é uma variação de um famoso sistema chamado C4.5, que foi descrito por Quinlan [56] que usa árvores de decisão para construir um classificador (WEKA inclusive nos fornece uma visão da árvore gerada com todos os seus pesos.

NaïveBayes é um método probabilístico que possui duas assunções: que os atributos são igualmente importantes e estatisticamente independente (essa assunção de indepen-

dência nunca é 100% correta, mas os métodos baseados nela geralmente funcionam bem na prática). Observando as métricas que iremos coletar e o conjunto de dados que será gerado após a aplicação da pesquisa, esse dois métodos parecem servir bem aos nossos propósitos.

Nós iremos usar também um terceiro algoritmo chamado AttributeSelectClassifier, que na verdade usa um método de seleção de características (no nosso caso, o GainRatio) e um algoritmo para executar a classificação (no nosso caso, J48 ou NaïveBayes). Esse é o jeito mais apropriado para aplicar seleção de características porque assim o algoritmo seleciona as características no conjunto de treinamento, tornando o resultados mais confiáveis.

Por fim, para conduzir todas essas análises nós iremos usar uma modo do programa WEKA chamado EXPERIMENTER, que nos permite rodar o mesmo experimento mais de uma vez e determinar a média e o desvio padrão, para evitar falsos resultados otimistas ou pessimistas (alta ou baixa acurácia) devido à seleção de atributos nos conjuntos de treinamento e teste. Esse modo também nos permite comprar o resultados de diferentes algoritmos.

Tabela 3 – Goal-Question-Metric

Objetivo	**Propósito**	Medir
	Questão	a importância
	Objeto	do desenvolvedor
	Ponto de vista	sob a perspectiva do supervisor
Pergunta	Qual a produtividade desse desenvolvedor?	
Métricas	Quantidade de entregas por mês	
	Avaliação subjetiva da produtividade	
Pergunta	Qual o nível de habilidade técnica que o desenvolvedor em questão possui?	
Métricas	Experiência relevante	
	Conhecimento especializado	
	Diversidade de habilidades	
	Resolução de problemas complexos	
Pergunta	Qual o nível de habilidade interpessoal que o desenvolvedor em questão possui?	
Métricas	Comunicação com colegas	
	Disposição para ajudar colegas quando solicitado	
Pergunta	Quando se depara com um problema, qual o principal comportamento do desenvolvedor?	
Métricas	(Introspectivo) Tenta resolver sozinho	
	(Introspectivo) Busca na documentação e livros	
	(Comunicativo) Procura ou pergunta em sites de perguntas e respostas	
	(Comunicativo) Pede ajuda para os colegas ou supervisores	
Pergunta	Qual o nível dessas características no perfil comportamental do desenvolvedor?	
Métricas	Liderança	
	Criatividade	
	Empreendedorismo	
	Pró-atividade	
Pergunta	Qual o compromisso do desenvolvedor em relação à empresa?	
Métricas	Tempo de trabalho	
	Organização e planejamento	
	Foco no cliente	
	Foco nos resultados	

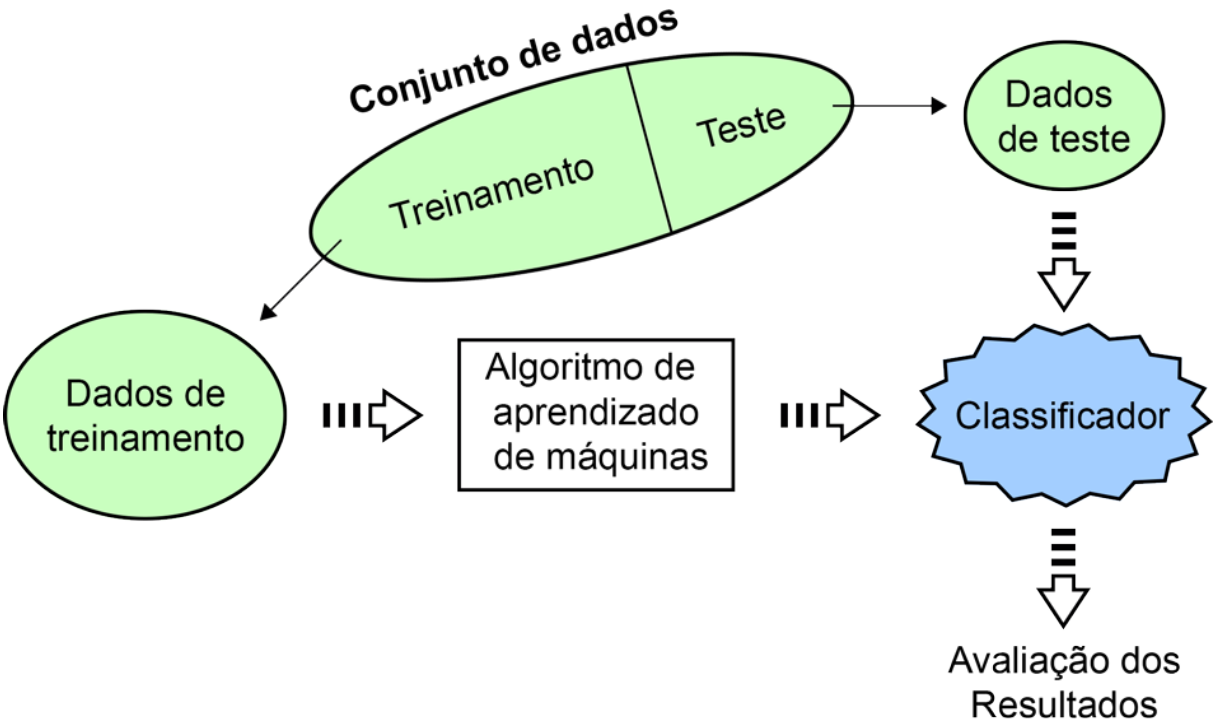

Figura 1 – Processo dos algoritmos de aprendizado de máquinas

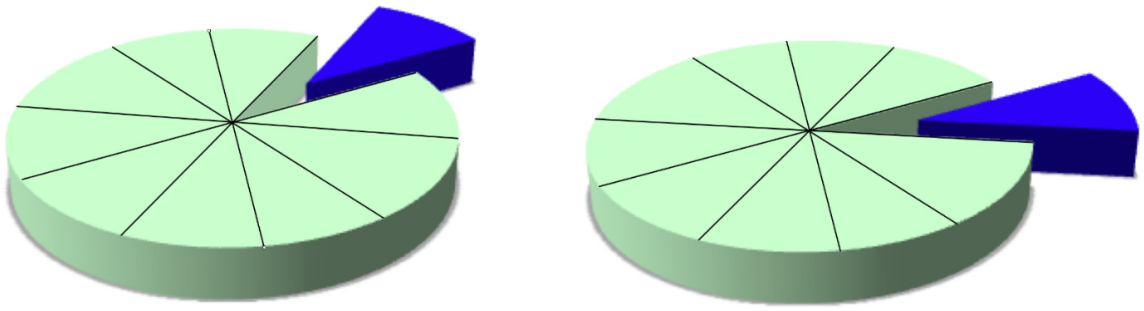

Figura 2 – Representação do 10-fold cross-validation

Capítulo 4

Resultados Gerais (Todas Empresas)

4.1 Considerações Iniciais

Seguindo os passos apresentados no capítulo anterior, nós iremos mostrar os resultados da aplicação da pesquisa, a seleção de características executada no conjunto de dados gerados por essa pesquisa e a aplicação dos classificadores e suas respectivas acurácias na classificação dos desenvolvedores.

4.2 Pesquisa

Primeiramente nós iremos apresentar os dados coletados com a pesquisa. Onze respondentes (supervisores) forneceram 61 respostas (avaliações únicas de desenvolvedores). Em alguns casos, alguns supervisores trabalham na mesma empresa, porém eles supervisionam diferente times. No total, oito empresas foram envolvidas na coleta de dados.

Nós pedimos para os supervisores classificarem os desenvolvedores em 5 graus de importância. Esses graus de importância e a distribuição dos desenvolvedores avaliados neles é mostrado na Figura 3.

Analisando os resultados da pesquisa, nós chegamos à conclusão que os supervisores foram, em certo nível, conservadores em classificar seus desenvolvedores nas classificações mais baixas de importância.

A partir dessa análise, juntamente com a análise da matriz de confusão gerada pela aplicação dos algoritmos descritos na seção 3.4 , nós decidimos agrupar os desenvolvedores em apenas duas classes, baseadas nas cinco classes originais de importância, como mostrado na Tabela 4, com o intuito de obter uma análise mais realística.

Utilizando esse novo conjunto de classes para agrupar os desenvolvedores, obtivemos a distribuição mostrada na Figura 4.

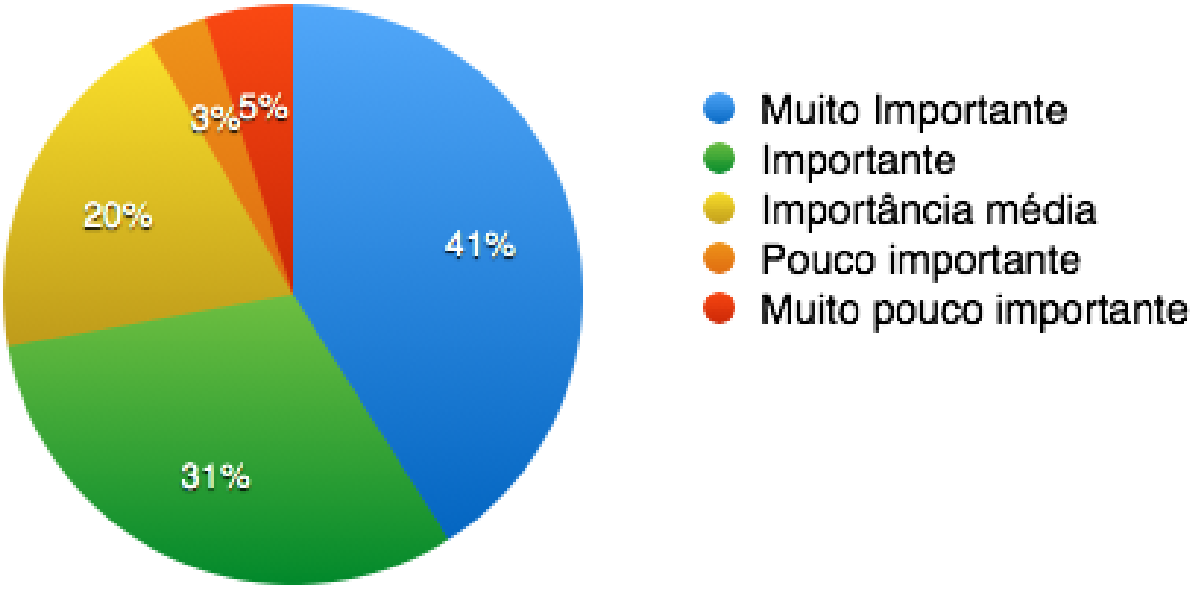

Figura 3 – Distribuição dos desenvolvedores avaliados pelas classes de importância

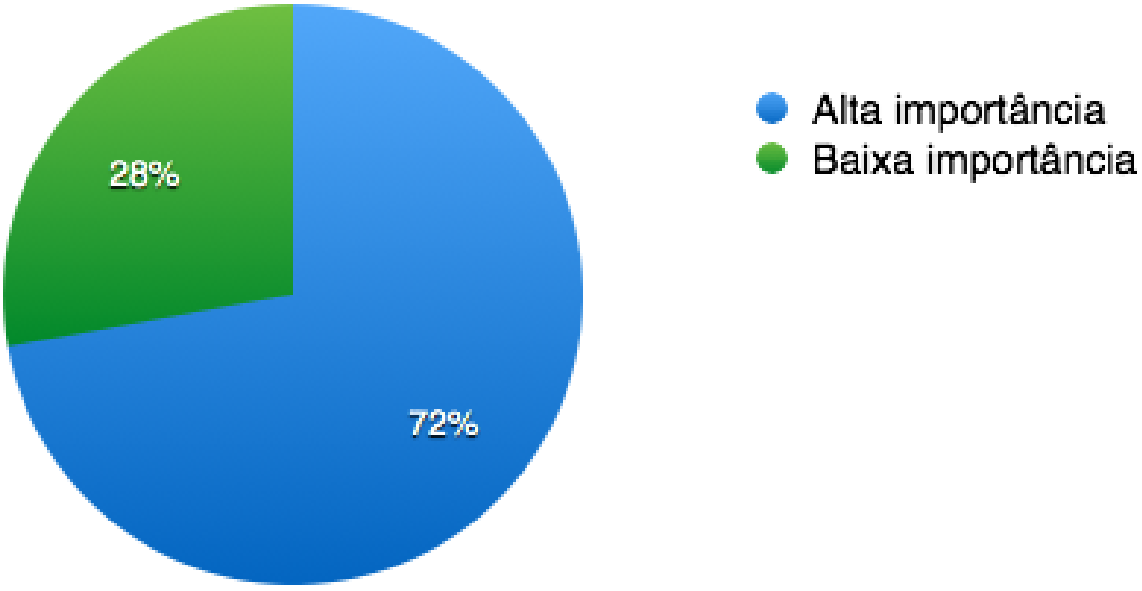

Figura 4 – Distribuição dos desenvolvedores avaliados pela nova classe de importância

Tabela 4 – Novo conjunto de classes de desenvolvedores

Novo conjunto de classes	**Conjunto de classes originais**
Alta importância	Muito importante
	Importante
Baixa importância	Importância média
	Pouco importante
	Muito pouco importante

4.3 Seleção de Características

Como explicado na seção 3.3 , nós usamos o algoritmo GainRatio para ordenar os atributos propostos, para prosseguirmos com a seleção de características. A Tabela 5 mostra as características ordenadas pelo mérito médio (um grau de influência do atributo na classificação, calculado pelo algoritmo) resultante da aplicação do algoritmo usando as classes originais e a Tabela 6 mostra a mesma visão, utilizando agora as novas classes.

Tabela 5 – Ordenação dos atributos (conjunto original de 5 classes)

Atributos	**Posição média**	**Mérito médio**
Capacidade de resolução de problemas complexos	1.4 +- 0.49	0.303 +- 0.03
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	1.6 +- 0.49	0.29 +- 0.022
Pró-atividade	3.6 +- 0.92	0.226 +- 0.01
Experiência relevante	4.8 +- 1.17	0.211 +- 0.014
Diversidade de habilidades	5.6 +- 1.36	0.202 +- 0.022
Conhecimento especializado	6.1 +- 2.51	0.2 +- 0.026
Tempo de trabalho (meses)	7.1 +- 1.37	0.184 +- 0.012
Criatividade	7.4 +- 1.62	0.18 +- 0.021
Foco nos resultados	8.7 +- 1.27	0.167 +- 0.017
Foco no cliente	10 +- 2.45	0.148 +- 0.023
PRINCIPAL comportamento do desenvolvedor	11.1 +- 1.58	0.14 +- 0.021
Comunicação com os colegas	11.6 +- 1.02	0.137 +- 0.011
Organização e planejamento	12.3 +- 1.27	0.122 +- 0.015
Liderança	14.5 +- 1.2	0.097 +- 0.018
Empreendedorismo	15 +- 0.89	0.087 +- 0.012
Disposição para ajudar colegas quando solicitado	15.2 +- 0.6	0.084 +- 0.014

Tabela 6 – Ordenação dos atributos (novo conjunto de classes)

Atributos	**Posição média**	**Mérito médio**
Pró-atividade	1.2 +- 0.4	0.168 +- 0.01
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	1.9 +- 0.54	0.156 +- 0.017
Capacidade de resolução de problemas complexos	3.2 +- 0.6	0.126 +- 0.013
Foco nos resultados	4.8 +- 0.98	0.112 +- 0.018
Experiência relevante	4.9 +- 1.3	0.107 +- 0.018
Criatividade	6.5 +- 1.36	0.095 +- 0.008
Organização e planejamento	6.9 +- 2.21	0.09 +- 0.014
Diversidade de habilidades	8.1 +- 1.14	0.08 +- 0.011
Conhecimento especializado	9.2 +- 1.78	0.071 +- 0.013
Foco no cliente	10.3 +- 1.95	0.063 +- 0.012
Tempo de trabalho (meses)	10.8 +- 1.08	0.063 +- 0.01
Disposição para ajudar colegas quando solicitado	11.6 +- 2.24	0.057 +- 0.019
Liderança	12 +- 0.89	0.052 +- 0.004
Comunicação com os colegas	13.6 +- 0.92	0.039 +- 0.009
Empreendedorismo	15.5 +- 0.5	0.015 +- 0.005
PRINCIPAL comportamento do desenvolvedor	15.5 +- 0.5	0.016 +- 0.007

4.4 Classificação

Considerando que os atributos foram ordenados, nós aplicamos a técnica de seleção de características e usamos apenas os atributos mais relevantes na classificação. Para determinar quantas características precisam ser selecionadas para obtermos uma maior performance, nós conduzimos um teste exaustivo (rodamos os classificadores com um crescente número de características selecionadas, de 2 a 16) e escolhemos a configuração com melhor performance (8 atributos). A página 45 e a Figura 6 mostram então a aplicação exaustiva do algoritmo que associa a seleção de características com os algoritmos de classificação, para o J48 e para o NaïveBayes respectivamente.

A Tabela 7 mostra os resultados da aplicação dos algoritmo J48. Como podemos ver, J48 não apresentou uma boa performance, com acurácia perto de 60%, para o conjunto original de classes, porém já apresentou uma acurácia significativa ao ser aplicado sobre o novo conjunto de classes (80% de acertos). No caso do J48, para ambos os conjuntos de classe, o classificador obteve a melhor performance utilizando apenas 2 atributos, como mostrado na Figura 5.

NaïveBayes, como mostrado na Tabela 8 obteve uma performance similar ao J48 quando aplicado no conjunto original de classes (acurácia perto de 61%). Já para o novo conjunto de classes, NaïveBayes atingiu uma acurácia expressiva de 85%. Ambos os melhores resultados do NaïveBayes foram atingido utilizando 8 características, como mostra a Figura 6.

Tabela 7 – Aplicação do J48 para os diferentes conjuntos de classe

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	60.64%
Novo conjunto de classes	80.14%

Tabela 8 – Aplicação do NaïveBayes para os diferentes conjuntos de classe

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	61.12%
Novo conjunto de classes	85.62%

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       08/12/14 13:30


Dataset                    (1) 'Conjunt | (2) 'Novo
-----------------------------------------------------
meta.AttributeSelectedCla(100)   60.64 |   80.14 v
meta.AttributeSelectedCla(100)   55.38 |   78.33 v
meta.AttributeSelectedCla(100)   58.55 |   77.67 v
meta.AttributeSelectedCla(100)   57.14 |   77.50 v
meta.AttributeSelectedCla(100)   56.31 |   77.02 v
meta.AttributeSelectedCla(100)   53.45 |   76.52 v
meta.AttributeSelectedCla(100)   51.88 |   75.88 v
meta.AttributeSelectedCla(100)   51.07 |   75.24 v
meta.AttributeSelectedCla(100)   50.60 |   74.74 v
meta.AttributeSelectedCla(100)   49.29 |   74.74 v
meta.AttributeSelectedCla(100)   48.64 |   74.74 v
meta.AttributeSelectedCla(100)   48.81 |   74.74 v
meta.AttributeSelectedCla(100)   48.14 |   74.74 v
meta.AttributeSelectedCla(100)   48.48 |   74.60 v
meta.AttributeSelectedCla(100)   48.48 |   74.60 v
-----------------------------------------------------
                              (v/ /*) |  (15/0/0)


Key:
(1) 'Conjunto de classes original'
(2) 'Novo conjunto de classes'
```

Figura 5 – Aplicação exaustiva do J48

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       08/12/14 13:33


Dataset                   (1) 'Conjunt | (2) 'Novo
----------------------------------------------------
meta.AttributeSelectedCla(100)   60.02 |   80.81 v
meta.AttributeSelectedCla(100)   57.19 |   80.95 v
meta.AttributeSelectedCla(100)   55.79 |   82.38 v
meta.AttributeSelectedCla(100)   57.88 |   82.83 v
meta.AttributeSelectedCla(100)   59.52 |   85.62 v
meta.AttributeSelectedCla(100)   58.81 |   84.69 v
meta.AttributeSelectedCla(100)   61.12 |   85.21 v
meta.AttributeSelectedCla(100)   58.50 |   82.55 v
meta.AttributeSelectedCla(100)   58.52 |   82.55 v
meta.AttributeSelectedCla(100)   59.21 |   82.21 v
meta.AttributeSelectedCla(100)   58.71 |   82.55 v
meta.AttributeSelectedCla(100)   59.02 |   83.55 v
meta.AttributeSelectedCla(100)   59.48 |   83.88 v
meta.AttributeSelectedCla(100)   58.69 |   84.17 v
meta.AttributeSelectedCla(100)   58.57 |   83.52 v
----------------------------------------------------
                              (v/ /*) |  (15/0/0)


Key:
(1) 'Conjunto de classes original'
(2) 'Novo conjunto de classes'
```

Figura 6 – Aplicação exaustiva do NaïveBayes

Capítulo 5

Resultados Individuais (Por Empresa)

5.1 Considerações Iniciais

Em nossa pesquisa, para as 8 empresas que participaram, 3 delas atingiram o número mínimo de respostas que permitem uma análise individual da empresa (10 avaliações de desenvolvedores). Nós iremos mostrar nesse capítulo o resultado que obtivemos analisando essas 3 empresas. Para proteger as informações dessas empresas, elas não serão identificadas. Nos referenciaremos à elas então como "Empresa A", "Empresa B" e "Empresa C".

5.2 Análise das Empresas

Para cada uma das 3 empresas, faremos a análise utilizando os dois conjuntos de classes, apontando as diferenças nos resultados. Mostraremos então a distribuição dos desenvolvedores ao longo dos dois conjuntos de classes, a seleção de características executadas no conjunto de dados resultantes da pesquisa respondida pelos supervisores das respectivas empresas e o resultado da aplicação dos algoritmos de classificação (dados de diferentes times, porém pertencentes à mesma empresa foram mesclados para obtermos uma visão geral da empresa).

Assim como fizemos com o conjunto de dados que representava todas as empresas, aplicamos aqui os algoritmos de classificação J48 e NaïveBayes nos dados fornecidos pelas respectivas empresas, utilizando tanto o conjunto original de classes quanto o novo conjunto de classes.

Como explicado na seção 3.4 , para associar a seleção de características com os algoritmos de classificação da maneira correta, utilizaremos o Algoritmo AttributeSelectedClassifier, usando o algoritmo GainRatioAttributeEval para selecionar os atributos mais relevantes. E da mesma maneira que foi realizado a seleção de características ao analisarmos o conjunto de dados de todas as empresas (apresentado na seção 4.4), a quantidade

de características ideal que maximiza a acurácia dos classificadores é escolhida através de um teste exaustivo, começando com apenas duas características e aumentando uma a uma até chegar nas dezesseis características apresentadas na Tabela 3 (já começamos com um número maior que 1 pois o objetivo não é encontrar correlação de alguma característica em específico com a classe, e sim encontrar um padrão na combinação de características que se correlacione com a importância dada pelo supervisor).

5.2.1 Análise da Empresa A

5.2.1.1 Pesquisa

A empresa A obteve avaliações de 2 supervisores. Como eram times com propósitos bem diferentes, consideramos apenas as avaliações de um dos times para representar a Empresa A, visto que o segundo não obteve avaliações suficientes para ser analisado individualmente. Logo, o time avaliado obteve 19 avaliações de desenvolvedores. A página 49 e a Figura 8 mostram a distribuição dos desenvolvedores pelo conjunto de classes original e pelo novo conjunto de classes, respectivamente.

5.2.1.2 Seleção de Características

O resultado da aplicação do algoritmo de seleção de características para o conjunto original de classes e para o novo conjunto de classes é apresentado na Tabela 9 e na Tabela 10 respectivamente.

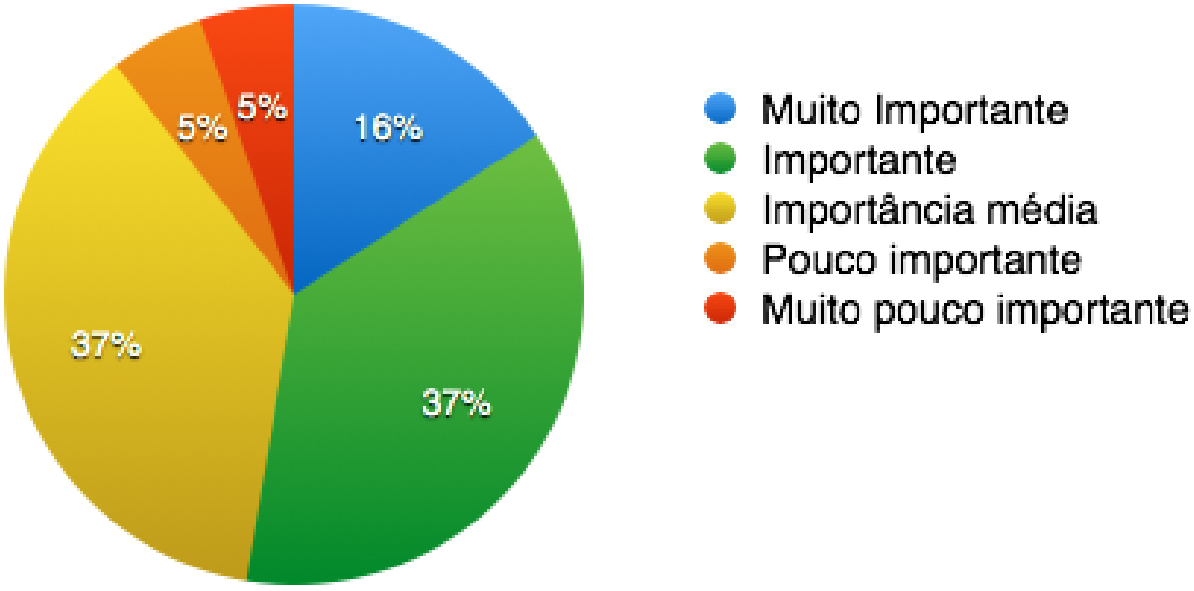

Figura 7 – Distribuição dos desenvolvedores pelo conjunto original de classes (Empresa A)

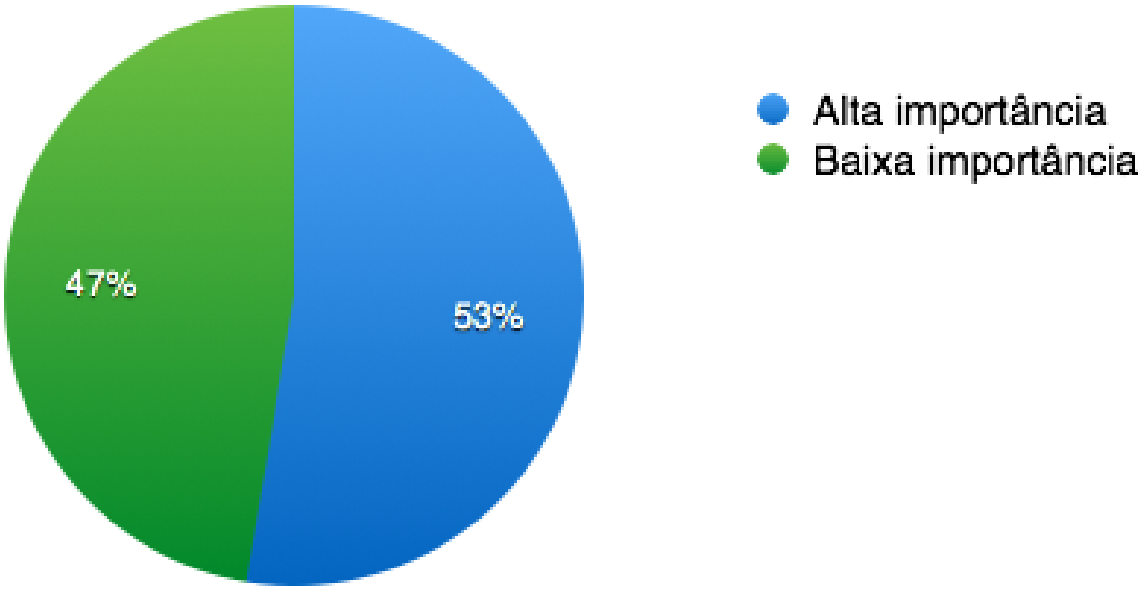

Figura 8 – Distribuição dos desenvolvedores pelo novo conjunto de classes (Empresa A)

Tabela 9 – Ordenação dos atributos da Empresa A (conjunto original de 5 classes)

Atributos	**Posição média**	**Mérito médio**
Capacidade de resolução de problemas complexos	1.5 +- 0.81	0.582 +- 0.038
Foco nos resultados	2.6 +- 1.02	0.519 +- 0.047
Criatividade	3 +- 1	0.504 +- 0.035
Pró-atividade	5.1 +- 2.12	0.456 +- 0.048
Foco no cliente	6.4 +- 1.96	0.437 +- 0.04
Diversidade de habilidades	6.5 +- 1.91	0.435 +- 0.025
Experiência relevante	6.9 +- 2.39	0.427 +- 0.035
Conhecimento especializado	8.9 +- 3.05	0.396 +- 0.053
PRINCIPAL comportamento do desenvolvedor	9.4 +- 2.91	0.384 +- 0.058
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	9.8 +- 2.44	0.383 +- 0.041
Comunicação com os colegas	10.5 +- 2.16	0.371 +- 0.051
Empreendedorismo	11.7 +- 1.79	0.351 +- 0.047
Disposição para ajudar colegas quando solicitado	12.1 +- 2.62	0.348 +- 0.05
Organização e planejamento	12.6 +- 2.24	0.336 +- 0.042
Tempo de trabalho (meses)	14.5 +- 4.5	0.064 +- 0.192
Liderança	14.5 +- 0.81	0.285 +- 0.063

Tabela 10 – Ordenação dos atributos da Empresa A (novo conjunto de classes)

Atributos	**Posição média**	**Mérito médio**
Pró-atividade	1.2 +- 0.4	0.323 +- 0.021
Capacidade de resolução de problemas complexos	2.4 +- 0.8	0.298 +- 0.032
Comunicação com os colegas	4 +- 1.48	0.254 +- 0.024
Foco nos resultados	5 +- 1.48	0.23 +- 0.029
Criatividade	6.7 +- 1.68	0.206 +- 0.025
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	7.5 +- 2.01	0.187 +- 0.026
Organização e planejamento	7.8 +- 3.22	0.191 +- 0.035
Conhecimento especializado	8.2 +- 3.28	0.189 +- 0.056
Experiência relevante	9.2 +- 3.12	0.173 +- 0.04
Empreendedorismo	10.2 +- 3.22	0.171 +- 0.037
PRINCIPAL comportamento do desenvolvedor	10.4 +- 3.8	0.172 +- 0.052
Disposição para ajudar colegas quando solicitado	11.1 +- 4.35	0.156 +- 0.049
Foco no cliente	11.2 +- 1.72	0.158 +- 0.02
Liderança	13 +- 2.53	0.137 +- 0.039
Diversidade de habilidades	13.4 +- 3.1	0.136 +- 0.046
Tempo de trabalho (meses)	14.7 +- 1.68	0.123 +- 0.022

5.2.1.3 Classificação

A Tabela 11 e a Tabela 12 apresentam os melhores resultados obtidos através da aplicação dos algoritmos J48 e NaïveBayes nos dados da Empresa A, respectivamente. Os algoritmos foram aplicados em ambos conjunto original de 5 classes e no novo conjunto de classes.

A Figura 9 e a Figura 10 mostram a aplicação exaustiva do algoritmo que associa a seleção de características com os algoritmos de classificação, para o J48 e para o NaïveBayes respectivamente.

Observando o teste exaustivo do J48, pudemos notar que ele obteve uma melhor performance utilizando um pequeno número de características ao realizar a classificação. No caso do conjunto original de 5 classes, os melhores resultados foram obtidos selecionando de 2 a 4 características, e ao ser aplicado utilizando o novo conjunto de classes, a melhor acurácia foi obtida selecionando apenas 2 características. É importante ressaltar também o aumento de aproximadamente 20% na acurácia do classificador utilizando a segunda classe de dados.

Diferentemente do J48, o NaïveBayes obteve uma melhor performance ao selecionar um maior número de características, de 13 a 15 no conjunto original de classes e acima de 7 no novo conjunto de classes. O NaïveBayes também teve uma melhoria de performance considerável ao utilizar o novo conjunto de dados (11%).

Tabela 11 – Aplicação do J48 para os diferentes conjuntos de classe da Empresa A

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	59%
Novo conjunto de classes	79.50%

Tabela 12 – Aplicação do NaïveBayes para os diferentes conjuntos de classe da Empresa A

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	68.50%
Novo conjunto de classes	79.50%

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       01/12/14 21:26


Dataset                    (1) 'Empresa | (2) 'Empr
-----------------------------------------------------
meta.AttributeSelectedCla(100)   57.00 |   79.50
meta.AttributeSelectedCla(100)   59.00 |   76.00
meta.AttributeSelectedCla(100)   59.00 |   75.50
meta.AttributeSelectedCla(100)   59.00 |   75.50
meta.AttributeSelectedCla(100)   57.00 |   75.50
meta.AttributeSelectedCla(100)   57.00 |   75.50
meta.AttributeSelectedCla(100)   57.00 |   75.50
meta.AttributeSelectedCla(100)   57.00 |   75.50
meta.AttributeSelectedCla(100)   56.50 |   75.50
meta.AttributeSelectedCla(100)   56.50 |   74.50
meta.AttributeSelectedCla(100)   56.50 |   74.50
meta.AttributeSelectedCla(100)   56.50 |   74.50
meta.AttributeSelectedCla(100)   56.50 |   74.50
meta.AttributeSelectedCla(100)   56.50 |   74.50
meta.AttributeSelectedCla(100)   56.50 |   74.50
-----------------------------------------------------
                               (v/ /*) |  (0/15/0)


Key:
(1) 'Empresa A (conjunto de classes original)'
(2) 'Empresa A (novo conjunto de classes)'
```

Figura 9 – Aplicação exaustiva do J48 para a Empresa A

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       01/12/14 21:19


Dataset                   (1) 'Empresa | (2) 'Empr
----------------------------------------------------
meta.AttributeSelectedCla(100)   58.00 |   78.00
meta.AttributeSelectedCla(100)   59.50 |   73.00
meta.AttributeSelectedCla(100)   55.50 |   68.00
meta.AttributeSelectedCla(100)   58.00 |   71.50
meta.AttributeSelectedCla(100)   57.00 |   75.50
meta.AttributeSelectedCla(100)   58.00 |   78.00
meta.AttributeSelectedCla(100)   59.00 |   79.50
meta.AttributeSelectedCla(100)   58.00 |   79.50
meta.AttributeSelectedCla(100)   64.00 |   79.50
meta.AttributeSelectedCla(100)   62.50 |   79.50
meta.AttributeSelectedCla(100)   67.00 |   79.50
meta.AttributeSelectedCla(100)   68.50 |   79.50
meta.AttributeSelectedCla(100)   68.50 |   79.50
meta.AttributeSelectedCla(100)   68.50 |   79.50
meta.AttributeSelectedCla(100)   55.50 |   79.50
----------------------------------------------------
                              (v/ /*) |  (0/15/0)


Key:
(1) 'Empresa A (conjunto de classes original)'
(2) 'Empresa A (novo conjunto de classes)'
```

Figura 10 – Aplicação exaustiva do NaïveBayes para a Empresa A

5.2.2 Análise da Empresa B

5.2.2.1 Pesquisa

A Empresa B obteve 10 avaliações de desenvolvedores feitas por um único supervisor. A Figura 11 e a Figura 12 mostram a distribuição dos desenvolvedores pelo conjunto de classes original e pelo novo conjunto de classes, respectivamente.

5.2.2.2 Seleção de Características

O resultado da aplicação do algoritmo de seleção de características para o conjunto original de classes e para o novo conjunto de classes é apresentado na Tabela 13 e na Tabela 14 respectivamente.

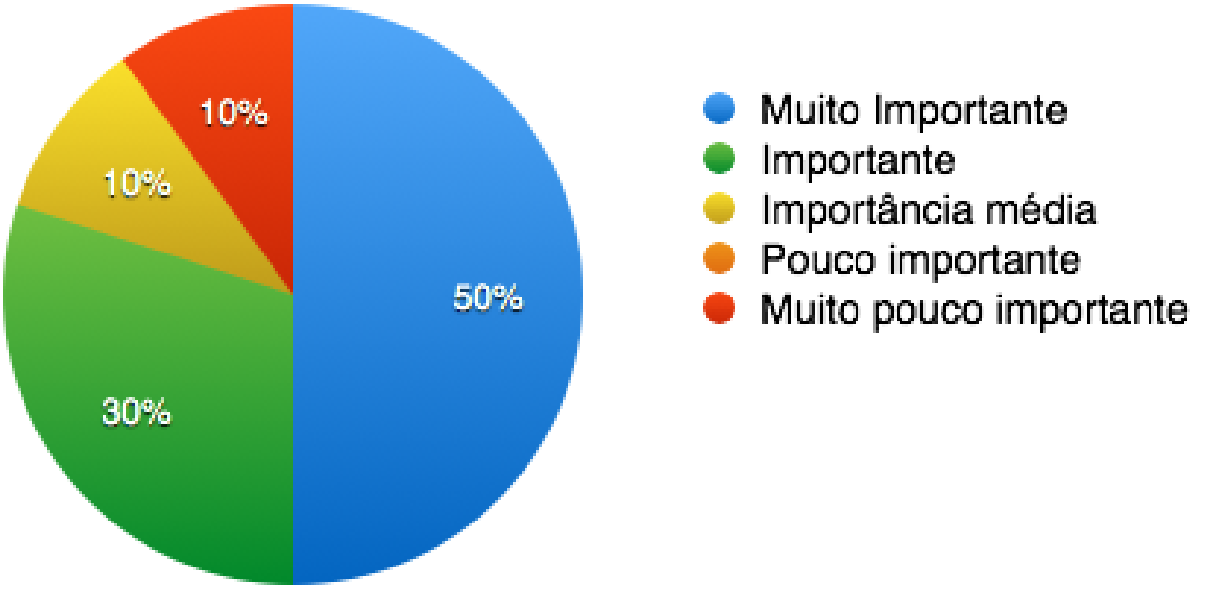

Figura 11 – Distribuição dos desenvolvedores pelo conjunto original de classes (Empresa B)

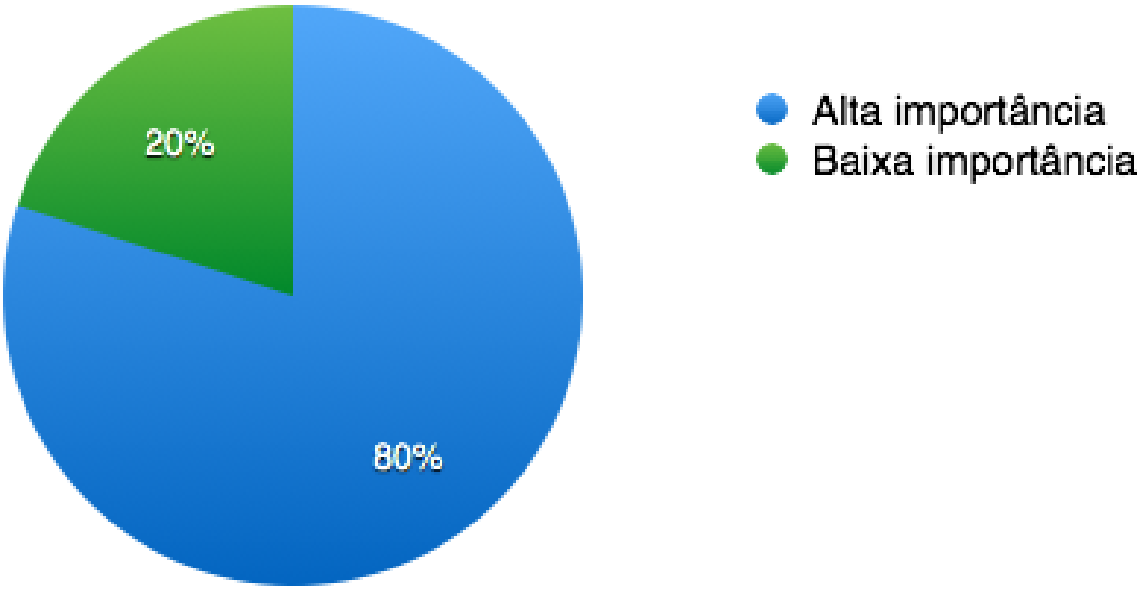

Figura 12 – Distribuição dos desenvolvedores pelo novo conjunto de classes (Empresa B)

Tabela 13 – Ordenação dos atributos da Empresa B (conjunto original de 5 classes)

Atributos	**Posição média**	**Mérito médio**
Capacidade de resolução de problemas complexos	1.3 +- 0.46	0.772 +- 0.078
Liderança	2.4 +- 1.02	0.652 +- 0.118
Tempo de trabalho (meses)	4.6 +- 1.43	0.562 +- 0.063
Foco nos resultados	5 +- 3.63	0.611 +- 0.179
Experiência relevante	5.3 +- 3.29	0.611 +- 0.179
Organização e planejamento	6.4 +- 1.8	0.506 +- 0.073
Conhecimento especializado	6.7 +- 2.1	0.515 +- 0.094
Diversidade de habilidades	8 +- 1.48	0.456 +- 0.06
Criatividade	9.7 +- 2.79	0.413 +- 0.119
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	11.1 +- 2.43	0.361 +- 0.076
Pró-atividade	11.2 +- 1.54	0.35 +- 0.04
Foco no cliente	11.2 +- 4.04	0.373 +- 0.133
Empreendedorismo	11.4 +- 1.74	0.36 +- 0.079
Disposição para ajudar colegas quando solicitado	12.4 +- 1.85	0.339 +- 0.08
Comunicação com os colegas	13.6 +- 2.15	0.318 +- 0.087
PRINCIPAL comportamento do desenvolvedor	15.7 +- 0.64	0.235 +- 0.048

Tabela 14 – Ordenação dos atributos da Empresa B (novo conjunto de classes)

Atributos	**Posição média**	**Mérito médio**
Foco nos resultados	2.9 +- 4.41	0.327 +- 0.131
Comunicação com os colegas	3.3 +- 1.42	0.244 +- 0.067
Experiência relevante	3.4 +- 4.25	0.327 +- 0.131
Criatividade	4.1 +- 2.07	0.26 +- 0.069
Capacidade de resolução de problemas complexos	5.6 +- 1.56	0.151 +- 0.052
Pró-atividade	8.1 +- 3.14	0.112 +- 0.028
PRINCIPAL comportamento do desenvolvedor	8.1 +- 3.18	0.112 +- 0.028
Tempo de trabalho (meses)	8.6 +- 2.29	0.109 +- 0.033
Diversidade de habilidades	8.6 +- 3.5	0.119 +- 0.06
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	9.4 +- 1.74	0.093 +- 0.022
Empreendedorismo	9.6 +- 1.85	0.093 +- 0.022
Disposição para ajudar colegas quando solicitado	11.3 +- 1.42	0.063 +- 0.02
Conhecimento especializado	12.7 +- 1.42	0.043 +- 0.026
Organização e planejamento	12.9 +- 2.84	0.031 +- 0.043
Foco no cliente	13.3 +- 3.23	0.031 +- 0.043
Liderança	14.1 +- 3.67	0.023 +- 0.04

5.2.2.3 Classificação

A Tabela 15 e a Tabela 16 apresentam os melhores resultados obtidos através da aplicação dos algoritmos J48 e NaïveBayes nos dados da empresa B, respectivamente. Os algoritmos foram aplicados em ambos conjunto original de 5 classes e no novo conjunto de classes.

A Figura 13 e a Figura 14 mostram a aplicação exaustiva do algoritmo que associa a seleção de características com os algoritmos de classificação, para o J48 e para o NaïveBayes respectivamente.

Observando o teste exaustivo do J48, pudemos notar que, para o conjunto de classes original, o algoritmo obteve uma melhor performance utilizando um menor número de características (de 2 a 6). Já ao utilizar o novo conjunto de classes, o classificador obteve a mesma acurácia independente do número de características utilizado. Utilizando o segundo conjunto de classes, o J48 teve um aumento de 10% em sua acurácia final.

No caso da Empresa B, o algoritmo NaïveBayes obteve uma performance extremamente semelhante ao J48, mantendo a performance praticamente constante de acordo com a variação das características selecionadas, obtendo uma melhoria de 10% ao utilizar o novo conjunto de classes, e obtendo uma acurácia final de 80%.

Tabela 15 – Aplicação do J48 para os diferentes conjuntos de classe da Empresa B

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	70%
Novo conjunto de classes	80%

Tabela 16 – Aplicação do NaïveBayes para os diferentes conjuntos de classe da Empresa B

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	70%
Novo conjunto de classes	80%

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       01/12/14 21:28


Dataset                   (1) 'Empresa | (2) 'Empr
----------------------------------------------------
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
meta.AttributeSelectedCla(100)   60.00 |   80.00
----------------------------------------------------
                               (v/ /*) |  (0/15/0)


Key:
(1) 'Empresa B (conjunto de classes original)'
(2) 'Empresa B (novo conjunto de classes)'
```

Figura 13 – Aplicação exaustiva do J48 para a Empresa B

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       01/12/14 21:29


Dataset                   (1) 'Empresa | (2) 'Empr
----------------------------------------------------
meta.AttributeSelectedCla(100)   70.00 |   70.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   70.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
meta.AttributeSelectedCla(100)   70.00 |   80.00
----------------------------------------------------
                              (v/ /*) |  (0/15/0)


Key:
(1) 'Empresa B (conjunto de classes original)'
(2) 'Empresa B (novo conjunto de classes)'
```

Figura 14 – Aplicação exaustiva do NaïveBayes para a Empresa B

5.2.3 Análise da Empresa C

5.2.3.1 Pesquisa

A Empresa C obteve 18 avaliação de desenvolvedores, dadas por 2 diferentes supervisores. Apesar de serem times diferentes, ambos fazem parte da mesma indústria de software, por isso resolvemos mesclar os dados e fazer uma avaliação conjunta. A Empresa C também obteve mais 2 avaliações dadas por um terceiro supervisor, mas os dados foram excluídos dessa análise pelo propósito desse time ser bastante diferente dos dois primeiros. A Figura 15 e a Figura 16 mostram a distribuição dos desenvolvedores pelo conjunto de classes original e pelo novo conjunto de classes, respectivamente.

5.2.3.2 Seleção de Características

O resultado da aplicação do algoritmo de seleção de características para o conjunto original de classes e para o novo conjunto de classes é apresentado na Tabela 17 e na Tabela 18 respectivamente.

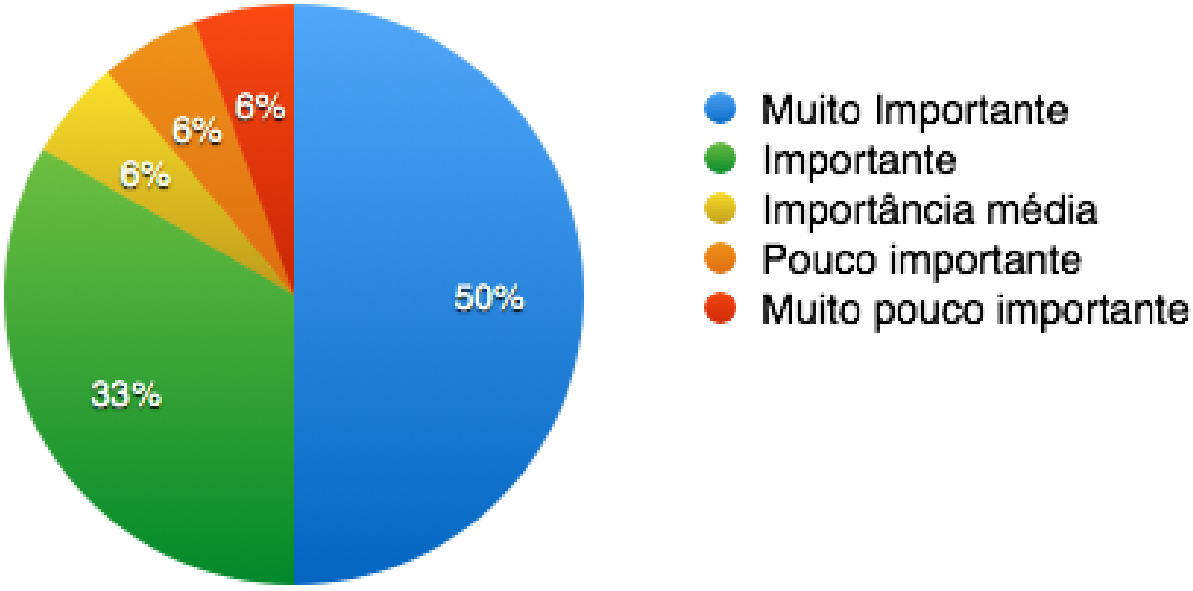

Figura 15 – Distribuição dos desenvolvedores pelo conjunto original de classes (Empresa C)

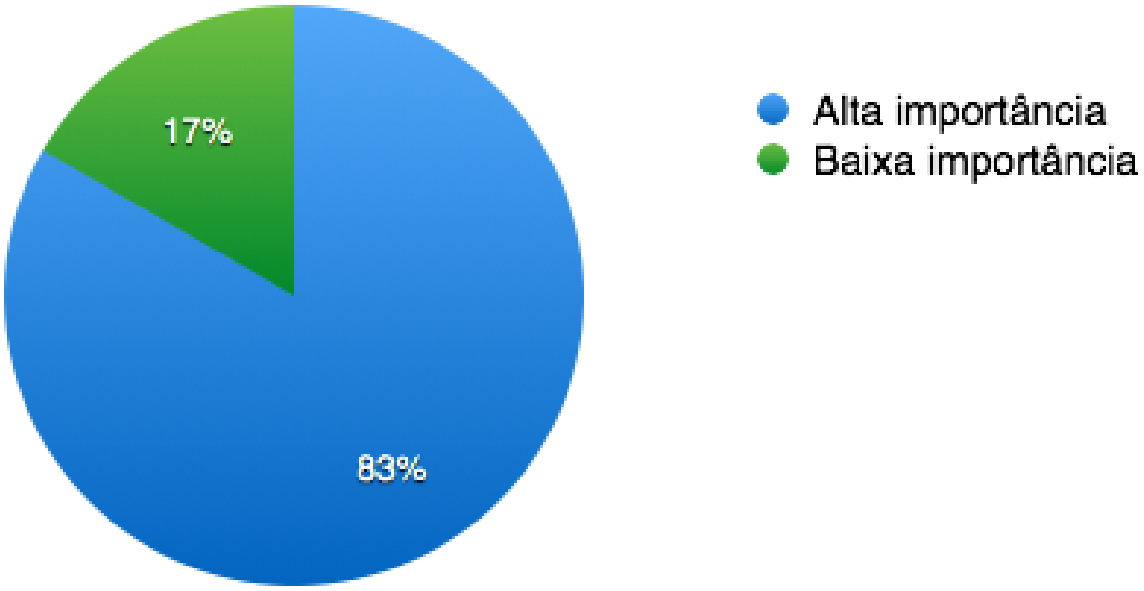

Figura 16 – Distribuição dos desenvolvedores pelo novo conjunto de classes (Empresa C)

Tabela 17 – Ordenação dos atributos da Empresa C (conjunto original de 5 classes)

Atributos	**Posição média**	**Mérito médio**
Capacidade de resolução de problemas complexos	1.1 +- 0.3	0.643 +- 0.044
Experiência relevante	2.2 +- 0.4	0.581 +- 0.037
Conhecimento especializado	3.3 +- 0.9	0.518 +- 0.058
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	3.4 +- 0.66	0.504 +- 0.036
Diversidade de habilidades	5.2 +- 0.6	0.386 +- 0.046
PRINCIPAL comportamento do desenvolvedor	6.4 +- 0.8	0.338 +- 0.048
Comunicação com os colegas	8.1 +- 1.45	0.301 +- 0.029
Foco nos resultados	8.5 +- 2.77	0.3 +- 0.08
Empreendedorismo	9.6 +- 2.06	0.275 +- 0.037
Pró-atividade	10.7 +- 2	0.267 +- 0.058
Tempo de trabalho (meses)	11.2 +- 1.66	0.261 +- 0.035
Foco no cliente	11.3 +- 1.68	0.255 +- 0.045
Criatividade	11.9 +- 2.55	0.246 +- 0.056
Liderança	13.6 +- 2.01	0.21 +- 0.055
Organização e planejamento	14.2 +- 1.54	0.196 +- 0.042
Disposição para ajudar colegas quando solicitado	15.3 +- 0.78	0.186 +- 0.044

Tabela 18 – Ordenação dos atributos da Empresa C (novo conjunto de classes)

Atributos	**Posição média**	**Mérito médio**
Conhecimento especializado	1 +- 0	0.26 +- 0.028
Experiência relevante	2.3 +- 0.46	0.24 +- 0.026
Qual a sua avaliação sobre a produtividade do desenvolvedor em questão?	3.1 +- 0.83	0.227 +- 0.035
Capacidade de resolução de problemas complexos	4 +- 0.45	0.216 +- 0.029
Diversidade de habilidades	5.6 +- 0.66	0.179 +- 0.027
Disposição para ajudar colegas quando solicitado	6 +- 1.55	0.172 +- 0.034
Foco nos resultados	7.1 +- 1.51	0.16 +- 0.03
Pró-atividade	8.1 +- 1.97	0.147 +- 0.026
Tempo de trabalho (meses)	8.7 +- 1	0.133 +- 0.024
Organização e planejamento	10 +- 1	0.115 +- 0.025
Foco no cliente	10.8 +- 1.4	0.091 +- 0.028
Criatividade	12 +- 1.41	0.076 +- 0.015
Comunicação com os colegas	13.2 +- 0.75	0.064 +- 0.011
Liderança	14.4 +- 1.62	0.048 +- 0.015
Empreendedorismo	14.5 +- 1.02	0.048 +- 0.015
PRINCIPAL comportamento do desenvolvedor	15.2 +- 1.08	0.041 +- 0.015

5.2.3.3 Classificação

A Tabela 19 e a Tabela 20 apresentam os melhores resultados obtidos através da aplicação dos algoritmos J48 e NaïveBayes nos dados da empresa C, respectivamente. Os algoritmos foram aplicados em ambos conjunto original de 5 classes e no novo conjunto de classes.

A Figura 17 e a Figura 18 mostram a aplicação exaustiva do algoritmo que associa a seleção de características com os algoritmos de classificação, para o J48 e para o NaïveBayes respectivamente.

Observando o teste exaustivo do J48, pudemos notar que ele obteve uma melhor performance utilizando apenas 2 características ao realizar a classificação utilizando o primeiro conjunto de classes. Já ao utilizar o novo conjunto de classes, o classificador obteve a mesma acurácia independente do número de características utilizado. Utilizando o segundo conjunto de classes, o J48 teve um aumento de 8% em sua acurácia final.

Diferentemente do J48, o NaïveBayes, ao utilizar o primeiro conjunto de classes obteve melhor performance selecionando um número mais intermediário de características (7). Já ao utilizar o novo conjunto de classes, o classificador atingiu sua acurácia máxima utilizando 2 ou 3 características, um número relativamente baixo. Nesse caso, o algoritmo obteve um ganho de performance de quase 15% se comparado com quando utilizou o conjunto original de classes.

Tabela 19 – Aplicação do J48 para os diferentes conjuntos de classe da Empresa C

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	77%
Novo conjunto de classes	85%

Tabela 20 – Aplicação do NaïveBayes para os diferentes conjuntos de classe da Empresa C

Classe	**Porcentagem de acertos**
Conjunto original de 5 classes	79.50%
Novo conjunto de classes	94%

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       01/12/14 21:33


Dataset                    (1) 'Empresa | (2) 'Empr
-----------------------------------------------------
meta.AttributeSelectedCla(100)   77.00 |   85.00
meta.AttributeSelectedCla(100)   74.00 |   85.00
meta.AttributeSelectedCla(100)   74.00 |   85.00
meta.AttributeSelectedCla(100)   74.00 |   85.00
meta.AttributeSelectedCla(100)   74.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
meta.AttributeSelectedCla(100)   75.00 |   85.00
-----------------------------------------------------
                               (v/ /*) |  (0/15/0)


Key:
(1) 'Empresa C (conjunto de classes original)'
(2) 'Empresa C (novo conjunto de classes)'
```

Figura 17 – Aplicação exaustiva do J48 para a Empresa C

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Percent_correct
Datasets:   15
Resultsets: 2
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       01/12/14 21:31


Dataset                   (1) 'Empresa | (2) 'Empr
----------------------------------------------------
meta.AttributeSelectedCla(100)   70.50 |   91.50
meta.AttributeSelectedCla(100)   65.00 |   94.00 v
meta.AttributeSelectedCla(100)   70.50 |   94.00
meta.AttributeSelectedCla(100)   75.50 |   90.50
meta.AttributeSelectedCla(100)   77.50 |   93.50
meta.AttributeSelectedCla(100)   79.50 |   92.50
meta.AttributeSelectedCla(100)   78.00 |   89.50
meta.AttributeSelectedCla(100)   61.50 |   89.50
meta.AttributeSelectedCla(100)   54.50 |   89.50 v
meta.AttributeSelectedCla(100)   56.00 |   89.50 v
meta.AttributeSelectedCla(100)   54.50 |   89.50 v
meta.AttributeSelectedCla(100)   55.50 |   93.50 v
meta.AttributeSelectedCla(100)   51.50 |   93.50 v
meta.AttributeSelectedCla(100)   54.50 |   91.50 v
meta.AttributeSelectedCla(100)   56.00 |   91.50 v
----------------------------------------------------
                              (v/ /*) |   (8/7/0)


Key:
(1) 'Empresa C (conjunto de classes original)'
(2) 'Empresa C (novo conjunto de classes)'
```

Figura 18 – Aplicação exaustiva do NaïveBayes para a Empresa C

Capítulo 6

Discussão

6.1 Considerações Iniciais

O primeiro ponto a ser considerado nessa discussão é a criação do novo conjunto de classes. Como mostrado na Tabela 3, baseado no conjunto original de classes , que possui 5 diferentes classes, nós agrupamos essas 5 classes em apenas 2, criando um novo conjunto de classes que provaram melhorar a performance de todos os algoritmos de classificação que aplicamos, tanto para o conjunto geral de dados, que continha dados de todas as empresas, como para o conjunto gerado pelas três empresas que atingiram o número mínimo para obter uma análise individual. Como pudemos observar na Tabela 4 e na Tabela 5, esse novo conjunto de classes não alterou significativamente a importância dos atributos para a classificação, assim, podemos concluir que essa nova configuração de classes preserva o sentido da classificação original feita pelos supervisores por causa da pequena variação na posição dos atributos na ordenação.

6.2 Resultados Gerais (Todas Empresas)

Vale a pena discutir as 3 primeiras características (mais importantes), que aparecem em ambas as ordenações. É importante mencionar que elas possuem uma correlação positiva com a classe, o que significa que quanto melhor a avaliação da características; melhor a posição na classificação de importância do desenvolvedor. Uma delas é a produtividade do desenvolvedor, sob o ponto de vista do supervisor; nesse caso, produtividade representa a quantidade de trabalho entregue. Isso não foi uma surpresa pois, como mencionamos na introdução desse trabalho, esse é a métrica clássica para avaliação de performance dos desenvolvedores. Por outro lado, as outras duas características trazem novas informações relevantes à discussão.

Capacidade de resolver problemas complexos nos leva à uma direção oposta apresentada pela métrica clássica (quantidade de entregas), porque geralmente faz com que

a produção possua uma menor taxa de outputs (LOC ou FP) sobre inputs (recursos, tempo) consumido.

Pró-atividade é na verdade uma característica de comportamento necessária em times envolvidos na solução de problemas complexos invés de sistemas canônicos onde as tarefas são mais previsíveis. Ao avaliarmos os dados utilizando o novo conjunto de classes, Pró-atividade é tida como a característica que mais influencia a avaliação dos supervisores sobre os desenvolvedores. A área de recursos humanos pode conduzir um melhor processo de contratação sabendo que os supervisores dos times de desenvolvimento de software avaliam essas características como requisitos fundamentais.

Sobre o resultado dos classificadores, cada um deles teve um ganho em sua acurácia em torno de 20% utilizando o novo conjunto de classes. Quando avaliando todas as empresas em conjunto, o que obteve a melhor performance foi o algoritmo NaïveBayes, com uma acurácia de 85,62%, o que nós consideramos como um resultado útil e de sucesso. O algoritmo J48 também obteve uma acurácia significativa, de 80%. O uso desses classificadores pode ajudar os supervisores a conduzir uma análise mais coerente do perfil do seu time e de cada desenvolvedor.

6.3 Resultados Individuais (Por Empresa)

Primeiramente, ao falarmos sobre a análise individual que aplicamos a cada empresa, é importante mencionar a distribuição dos desenvolvedores pelas classes de importância. A Empresa A possui uma boa distribuição dos desenvolvedores, tanto para o conjunto de classes original, tanto para o novo conjunto de classes. Não acreditamos que nossa hipótese de que os supervisores possam ter ficados receosos no momento da classificação, que baseia a criação do novo conjunto de classes, se aplique a esse caso.

Diferentemente da Empresa A, as empresas B e C já não possuem uma boa distribuição dos desenvolvedores, por ambos os conjuntos de importância. Acreditamos que a nossa hipótese que baseia a criação do novo conjunto de classes se aplique bem a esses casos, porém mesmo com o agrupamento das classes de importância, obtivemos uma distribuição muito tendenciosa à classe de maior importância (cerca de 80% dos desenvolvedores dessas empresas são considerados como sendo de “Alta importância”). Dessa forma, como possuímos apenas duas classes no novo conjunto de classes, estatisticamente falando, se o nosso classificador julgasse todos os desenvolvedores como sendo de alta importância, ele teria uma acurácia de cerca de 80%. Logo, para considerarmos o resultado um sucesso, o classificador, ao utilizar o segundo conjunto de classes, deve apresentar uma acurácia de no mínimo 80%.

Quando analisamos as características que mais influenciam a avaliação dos supervisores (considerando apenas a ordenação do utilizando o novo conjunto de classes que se provou ser mais eficaz), por cada empresa, notamos algumas grandes diferenças, tanto

entre as empresa e em relação ao conjunto geral de dados (todas as empresas). Para todas as empresas, encontramos pelo menos uma característica técnica entre as top três características mais importantes. Nas empresas A e B, foram encontradas características comportamentais (Pró-atividade), de habilidade interpessoal (Comunicação com os colegas) e de compromisso em relação à empresa (Foco no resultado) com melhor posicionamento que a métrica clássica de produtividade (Quantidade de entregas).

Já a Empresa C preza mais por características técnicas, visto que dentre as top 5 primeiras características estão as 4 características técnicas e a métrica clássica de produtividade. Entendemos que essas diferenças nos resultados são devidas às diferenças na cultura e nos valores de cada empresa, vários estudos já foram feito sobre como a cultura corporativa pode interferir na produtividade dos desenvolvedores [13], [14], [18], [42], [57]–[59].

Em relação à classificação, todas as empresas obtiverem um aumento de 10% a 15% na acurácia dos classificadores ao utilizar o novo conjunto de classes de importância. A Empresa A, concidentemente, obteve uma acurácia de 79.5% para ambos os algoritmos J48 e NaïveBayes. A Empresa B, atingiu uma acurácia também igual entre os classificadores, de 80%, o que não foi considerado um bom resultado devida às condições mencionadas anteriormente sobre a distribuição dos desenvolvedores. A Empresa C por sua vez foi a que obteve a maior acurácia, utilizando o algoritmo NaïveBayes, de 94%. Devida à sua condição, similar à da Empresa B, consideramos esse resultado um sucesso (o algoritmo J48 também obteve uma acurácia superior ao limite estipulado, de 85%).

6.4 Ameaças À Validade

Por fim, é importante apontar algumas ameaças à validade desse estudo. O número limitado de desenvolvedores e empresas participantes nesse estudo, e o fato de todas estarem estabelecidas na mesma cidade podem limitar a generalização dos resultados para outros contextos. Apesar disso, observamos várias intersecções no resultado de diferentes empresas, o que pode mitigar parte desse risco. A classificação inicial de importância proporcionado pelos supervisores tendem a serem mais positivas, talvez porque eles não gostariam de dizer que mantém desenvolvedores com baixa importância em seus times. Porém a nova classificação proposta mitiga parte desse risco.

Capítulo 7

Conclusões

7.1 Considerações

Nós identificamos, como mostrado na seção 1.2 uma situação-problema nas empresas atualmente, que é a subjetividade na avaliação dos desenvolvedores por parte dos supervisores. Esse problema leva as empresas a terem problemas maiores, de retenção e motivação dos funcionários. O nosso estudo se propôs então a ajudar as empresas e, se possível fornecer uma ferramenta que auxilia e empresa a enfrentar com mais eficácia esses problemas.

Seguimos o estudo com base em duas principais perguntas. A primeira questionava quais seriam os critérios mais importantes utilizados pelos supervisores ao realizarem sua avaliação sobre os desenvolvedores. Para isso, primeiramente, precisamos levantar uma série de critérios e optar por aqueles que aparentassem estar mais ligados à realidade das empresas de hoje. Fizemos esse levantamento e chegamos a 16 métricas, divididas em 4 diferentes grupos, como mostrado na seção 2.2 . Utilizamos a ajuda do framework GQM para tal. Uma vez com as métricas levantadas, utilizamos algoritmos de seleção de características que ordenaram as 16 métricas por ordem de influência na classe. Então consideramos que respondemos com sucesso a primeira pergunta realizada.

A segunda pergunta questionava se seria possível atingir um classificador de alta acurácia, utilizando os critérios propostos, tanto para o conjunto que reuniam os dados de todas as empresas, quanto para cada empresa que atingisse o mínimo de dados necessários para uma análise individual. Como mostrado no Capítulo 4 e no Capítulo 5, consideramos também ter respondido com sucesso essa pergunta, visto que conseguimos um classificador com acurácia maior que 85% para o conjunto de dados de todas as empresas, e para as 3 empresas que se qualificaram para uma análise individual, os melhores classificadores obtiveram uma acurácia de cerca de 80%, sendo que uma empresa obteve um classificador com uma acurácia de 94%.

Nesse estudo nós proporcionamos então um conjunto de critérios utilizados pelos supervisores de empresas de T.I, para avaliar seus desenvolvedores, além de ordenarmos esses

critérios pela taxa de influência na classificação de importância, provendo uma análise em quais são os fatores mais relevantes.

Além disso, nós criamos um classificador de alta acurácia, que pode ajudar, por exemplo, os gerentes de recursos humanos a procurarem candidatos que possuam as características necessárias e que consequentemente possuam um bom potencial de se tornarem parte importante do time, e também pode auxiliar o supervisora realizar uma avaliação muito mais concreta, saindo da área da subjetividade no momento de avaliar a importância dos seus desenvolvedores.

7.2 Trabalhos Futuros

Obtivemos bons resultados com esse estudo sobre a avaliação da importância dos desenvolvedores sob a ótica do supervisor. Porém entendemos que muito mais pode ser realizado, pois lidamos com um problema significativo e a pesquisa também possui diversas áreas a serem ainda exploradas. Abaixo listamos algumas das possibilidades que encontramos:

1. Expandir a aplicação da pesquisa, aplicando em mais empresas situadas em diferentes regiões do país e até do mundo, bem como expandir as características utilizadas;

2. Realizar uma análise qualitativa, em cima dos dados de cada empresa, considerando como a cultura e os valores da empresa influenciam na avaliação de cada critério;

3. Aplicar esse classificador em colaboradores de repositórios de software open-source. Vasilescu et al. [60] realizou um estudo buscando encontrar relação entre a busca de conhecimento em sites de perguntas e respostas e a produtividade do desenvolvedor, porém utilizou a métrica clássica de produtividade, nesse caso, número de commits. Utilizar o nosso estudo as métricas que levantamos como base para validar os resultados ou mesmo detectar e avaliar as diferenças, pode ser uma sugestão de trabalho futuro para ambos os estudos.

Por se tratar de um problema importante para todas as empresas atualmente, entendemos que, além das possibilidades propostas, existem mais inúmeras possibilidades de trabalhos futuros que poderiam usar esse nosso estudo como base.

Referências

ABDEL-HAMID, T.; MADNICK, S. E. **Software Project Dynamics: An Integrated Approach**. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1991. ISBN 0-13-822040-9.

ALPER, S.; TJOSVOLD, D.; LAW, K. S. Conflict Management, Efficacy, and Performance in Organizational Teams. **Personnel Psychology**, v. 53, p. 625–642, 2000. ISSN 0031-5826. Disponível em: <http://doi.wiley.com/10.1111/j.1744-6570.2000.tb00216.x>.

BANKER, R. D.; DATAR, S. M.; KEMERER, C. F. Factors affecting software maintenance productivity: An exploratory study. In: **Proceedings of the International Conference on Information Systems**. Pittsburgh, PA: [s.n.], 1987. p. 160–175.

_____. **A Model to Evaluate Variables Impacting the Productivity of Software Maintenance Projects**. 1991. 1–18 p.

BOEHM, B. W. Software Engineering Economics. **IEEE Transactions on Software Engineering**, SE-10, 1984. ISSN 0098-5589.

_____. Improving Software Productivity. **Computer**, IEEE Computer Society, Los Alamitos, CA, USA, v. 20, n. 9, p. 43–57, 1987. ISSN 0018-9162.

BOEHM, B. W. et al. **Software Cost Estimation with Cocomo II with Cdrom**. 1st. ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2000. ISBN 0130266922.

_____. The TRW Software Productivity System. In: **Proceedings of the 6th International Conference on Software Engineering**. Los Alamitos, CA, USA: IEEE Computer Society Press, 1982. (ICSE '82), p. 148–156. Disponível em: <http://dl.acm.org/citation.cfm?id=800254.807757>.

BOEHM, B. W.; PAPACCIO, P. N. Understanding and controlling software costs. **Software Engineering, IEEE Transactions on**, v. 14, n. 10, p. 1462–1477, 1988. ISSN 0098-5589.

BRIAND, L.; EMAM, K. E.; BOMARIUS, F. COBRA: a hybrid method for software cost estimation, benchmarking, and risk assessment. **Proceedings of the 20th International Conference on Software Engineering**, 1998. ISSN 0270-5257.

Brooks Jr, F. P. No silver bullet-essence and accidents of software engineering. **IEEE computer**, v. 20, n. 4, p. 10–19, 1987.

BROOKS, W. Software Technology Payoff: Some Statistical Evidence. **J. Syst. Softw.**, Elsevier Science Inc., New York, NY, USA, v. 2, n. 1, p. 3–9, 1981. ISSN 0164-1212. Disponível em: <http://dx.doi.org/10.1016/0164-1212(81)90041-8>.

CALOW, H. Maintenance productivity factors-a case study. In: **Software Maintenance, 1991., Proceedings. Conference on**. [S.l.: s.n.], 1991. p. 250–253.

CHATZOGLOU, P. D.; MACAULAY, L. A. **The importance of human factors in planning the requirements capture stage of a project**. 1997. 39–53 p.

CHIAVENATO, I. **Gestão de pessoas**. 3ª. ed. [S.l.: s.n.], 2008.

CLINCY, V. A. Software Development Productivity and Cycle Time Reduction. **J. Comput. Sci. Coll.**, Consortium for Computing Sciences in Colleges, USA, v. 19, n. 2, p. 278–287, 2003. ISSN 1937-4771. Disponível em: <http://dl.acm.org/citation.cfm?id=948785.948824>.

COLE, A. Runaway Projects - Cause and Effects. **Software World (UK)**, v. 26, n. 3, 1995.

CORAM, M.; BOHNER, S. The impact of agile methods on software project management. In: **Engineering of Computer-Based Systems, 2005. ECBS '05. 12th IEEE International Conference and Workshops on the**. [S.l.: s.n.], 2005. p. 363–370.

de Barros Sampaio, S. C. et al. A Review of Productivity Factors and Strategies on Software Development. In: **Software Engineering Advances (ICSEA), 2010 Fifth International Conference on**. [S.l.: s.n.], 2010. p. 196–204.

DEMARCO, T.; TIMOTHY, L. **peopleware - productive projects and teams**. [S.l.: s.n.], 1987.

DUTRA, J. S. **Competências: Conceitos e Instrumentos para a Gestão de Pessoas na Empresa Moderna**. 1ª. ed. [S.l.: s.n.], 2004.

Faria Sueli, O. V. F. d. F. L. . D. F. Competências do profissional da informação: uma reflexão a partir da Classificação Brasileira de Ocupações. **Ciência da Informação**, scielo, v. 34, p. 26–33, 2005. ISSN 0100-1965. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-19652005000200003&nrm=iso>.

FINNIE, G. R.; WITTIG, G. E.; PETKOV, D. I. **Prioritizing software development productivity factors using the analytic hierarchy process**. 1993. 129–139 p.

FLEURY, M. T. L.; FLEURY, A. **Construindo o conceito de competência**. 2001. 183–196 p.

GUZZO, R. A. Productivity research: Reviewing psychological and economic perspectives. In: . [S.l.]: Jossey-Bass Publishers, 1988. p. 63–81.

HANSON, S. J.; ROSINSKI, R. R. Programmer Perceptions of Productivity and Programming Tools. **Commun. ACM**, ACM, New York, NY, USA, v. 28, n. 2, p. 180–189, 1985. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/2786.2791>.

HANTOS, P.; GISBERT, M. Identifying software productivity improvement approaches and risks: Construction industry case study. **IEEE Software**, v. 17, p. 48–56, 2000. ISSN 07407459.

HOLMES, G.; DONKIN, A.; WITTEN, I. H. W EKA : A Machine Learning Workbench.

JONES, C. **Programming Productivity: Steps Toward a Science. McGraw-Hill Series in Software Engineering and Technology.** [S.l.]: McGraw-Hill, 1986.

_____. **Applied software measurement: assuring productivity and quality**. [s.n.], 1997. ISBN 0070328137. Disponível em: <http://books.google.ch/books/about/Applied_software_measurement.html?id=U9JQAAAAMAAJ&redir_esc=y>.

_____. **Software Assessments, Benchmarks, and Best Practices**. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000. ISBN 0-201-48542-7.

LAKHANPAL, B. **Understanding the factors influencing the performance of software development groups: An exploratory group-level analysis**. 1993. 468–473 p.

LALSING, V.; KISHNAH, S.; PUDARUTH, S. People Factors in Agile Software Development and Project Management. **International Journal of Software Engineering & Applications**, v. 3, p. 117–138, 2012. Disponível em: <http://airccse.org/journal/ijsea/papers/3112ijsea09.pdf>.

LOKAN, C. J. Impact of Subjective Factors on Software Productivity. In: **Proceedings of 7th Australian Conference on Software Metrics**. Melbourne: [s.n.], 2001.

MAXWELL, K. D.; FORSELIUS, P. Benchmarking software development productivity. **Software, IEEE**, v. 17, p. 80–88, 2000. ISSN 0740-7459.

MELO, C. et al. Agile Team Perceptions of Productivity Factors. In: **Agile Conference (AGILE), 2011**. [S.l.: s.n.], 2011. p. 57–66.

RASCH, R. H. An Investigation of Factors That Impact Behavioral Outcomes of Software Engineers. In: **Proceedings of the 1991 Conference on SIGCPR**. New York, NY, USA: ACM, 1991. (SIGCPR '91), p. 38–53. ISBN 0-89791-389-2. Disponível em: <http://doi.acm.org/10.1145/111084.111090>.

SCACCHI, W.; HURLEY, D. Understanding software productivity. In: **Software Engineering and Knowledge Engineering: Trends for the Next Decade**. [s.n.], 1995. v. 4, p. 273–316. Disponível em: <http://books.google.com/books?hl=en&lr=&id=bDlZa4KvFGgC&oi=fnd&pg=PA273&dq=Understanding+software+productivity&ots=TFYqI5HTtY&sig=QBz5suT4_1Fwjkp-NwO0ZMeGEIY>.

SCHWABER, K. **Agile Project Management With Scrum**. Redmond, WA, USA: Microsoft Press, 2004. ISBN 073561993X.

SCUDDER, R. A.; KUCIC, A. R. Productivity Measures for Information Systems. **Inf. Manage.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 20, n. 5, p. 343–354, 1991. ISSN 0378-7206. Disponível em: <http://dx.doi.org/10.1016/0378-7206(91)90033-X>.

SHARP, H. et al. Models of Motivation in Software Engineering. **Inf. Softw. Technol.**, Butterworth-Heinemann, Newton, MA, USA, v. 51, n. 1, p. 219–233, 2009. ISSN 0950-5849. Disponível em: <http://dx.doi.org/10.1016/j.infsof.2008.05.009>.

SHARPE, J. L.; CANGUSSU, J. W. A productivity metric based on statistical pattern recognition. In: **Computer Software and Applications Conference, 2005. COMPSAC 2005. 29th Annual International**. [S.l.: s.n.], 2005. v. 1, p. 59–64 Vol. 2. ISSN 0730-3157.

SYMONS, C. Software Industry Performance: What You Measure Is What You Get. **Software, IEEE**, v. 27, n. 6, p. 66–72, 2010. ISSN 0740-7459.

TURCOTTE, J.; RENNISON, L. W. The Link between Technology Use, Human Capital, Productivity and Wages: Firm-level Evidence. **International Productivity Monitor**, v. 9, p. 25–36, 2004. Disponível em: <http://ideas.repec.org/a/sls/ipmsls/v9y20043.html>.

VOSBURGH, J. et al. Productivity factors and programming environments. In: **ICSE '84 Proceedings of the 7th International Conference on Software Engineering**. [s.n.], 1984. p. 143–152. ISBN 0-8186-0528-6. Disponível em: <http://dl.acm.org/citation.cfm?id=800054.801963>.

WAGNER, S.; RUHE, M. **A Structured Review of Productivity Factors in Software Development Technical** . [S.l.], 2008.

WALLACE, L.; KEIL, M.; RAI, A. Understanding Software Project Risk: A Cluster Analysis. **Inf. Manage.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 42, n. 1, p. 115–125, 2004. ISSN 0378-7206. Disponível em: <http://dx.doi.org/10.1016/j.im.2003.12.007>.

WALSTON, C. E.; FELIX, C. P. **A method of programming measurement and estimation**. 1977. 54–73 p.

WITTEN, I. H.; FRANK, E. **Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005. ISBN 0120884070.

WOHLIN, C.; AHLGREN, M. Soft factors and their impact on time to market. **Software Quality Journal**, v. 4, p. 189–205, 1995. ISSN 09639314.

WOHLIN, C.; ANDREWS, A. A. Assessing Project Success Using Subjective Evaluation Factors. **Software Quality Journal**, v. 9, p. 43–70, 2001. ISSN 09639314.

YU, W. D.; SMITH, D. P.; HUANG, S. T. Software productivity measurements. In: **Computer Software and Applications Conference, 1991. COMPSAC '91., Proceedings of the Fifteenth Annual International**. [S.l.: s.n.], 1991. p. 558–564.