10º Encontro de Ensino Pesquisa e Extensão

Patrocínio, MG, outubro de 2023

EXPLORAÇÃO DE ATRIBUTOS INFLUENTES NO DESEMPENHO DE EQUIPES DE DESENVOLVIMENTO DE SOFTWARE GEOGRAFICAMENTE DISTRIBUÍDAS

Kelvin Wesley Cunha de Faria, Danielli Araújo Lima
Instituto Federal do Triangulo Mineiro (IFTM) Campus Patrocínio
kelvinwesley90@hotmail.com, danielli@iftm.edu.br
Modalidade: Pesquisa
Formato: Artigo Completo

Resumo: Na gestão remota de equipes de desenvolvimento de software, a tecnologia nos permite transformar desafios em oportunidades, e este artigo explora o desempenho de algoritmos de classificação na previsão do desempenho de equipes de engenharia de software. Utilizando um conjunto de dados com 85 atributos, incluindo métricas de equipe e características individuais, aplicamos os algoritmos Decision Tree Learner, Gradient Boosted Learner, Naive Bayes Learner, Random Forest Learner e Support Vector Machine para avaliar sua precisão, sensibilidade e outras métricas. O GBT obteve a maior precisão de 91,17%, embora seja computacionalmente intensivo, enquanto o RFL alcançou 90,42% de precisão ao considerar apenas sete atributos, destacando a importância da seleção de atributos no desempenho do modelo. Este estudo enfatiza o equilíbrio entre precisão e eficiência computacional, identificando atributos-chave relevantes para a gestão de equipes de desenvolvimento de software remotas.
Palavras-chaves: Mineração de Dados, Aprendizado de Máquina, Inteligência Artificial, Engenharia de Software, Gestão de projetos.

Introdução

A informatização das empresas é realidade em todo o mundo. Seja para estar de acordo com gestões governamentais, para estar à frente da concorrência ou por aumento da eficiência e diminuição do tempo de execução de serviços, a Tecnologia da Informação (TI) é imprescindível aos mais diversos ramos. E como qualquer mudança, há sempre a capacitação e adequação ao novo cenário, ou seja, profissionais devem aperfeiçoar-se

e saber lidar com situações inovadoras e adversas. “Com o uso das novas tecnologias da informação e da comunicação, os trabalhadores não precisam permanecer fisicamente dentro das organizações, isso proporciona mais autonomia na escolha do local onde podem desenvolver o seu trabalho, possibilitando a flexibilização dos horários de trabalho” (TASCHETTO; FROEHLICH, 2019), modelo também conhecido hoje como "Home Office".

Esse modo de trabalho tem sido amplamente explorado, pela facilidade de acesso a profissionais capacitados de qualquer lugar do mundo, ganhando força com o início da pandemia do Covid-19, “impondo severas restrições à mobilidade e intensificando a adoção de novas práticas em termos de organização do trabalho” (PERDIGAO; SEDA et al., 2022), na qual diversos profissionais tiveram que desempenhar seus trabalhos de casa, para evitar contato com pessoas e propagação do vírus. E como se trata de um avanço em conjunto com a tecnologia, as empresas de TI tomam a frente na expansão desse conceito, principalmente no setor de desenvolvimento de software e aplicações.

Um ponto que merece reflexão é a eficácia das equipes de profissionais em regime de “Home Office”, distribuídas geograficamente. É importante entender se essas equipes conseguem alcançar um bom desempenho nas atividades designadas e identificar os fatores essenciais que contribuem para o sucesso no trabalho realizado dessa maneira. Assim, o objetivo desta pesquisa é examinar minuciosamente os elementos que influenciam o êxito das equipes de desenvolvimento de software que operam a partir de diferentes localizações geográficas, com foco na consecução de objetivos e metas. A técnica para identificar esses atributos envolverá o uso de técnicas de inteligência artificial, filtros de correlação e estatística descritiva de forma quantitativa. Por fim, os resultados serão analisados e discutidos de maneira qualitativa, com o intuito de auxiliar as empresas na formação de equipes de desenvolvimento de software geograficamente distribuídas de alta qualidade.

Fundamentação Teórica

A gestão eficaz de equipes de desenvolvimento de software é crucial na formação de engenheiros de software e na gestão de projetos. No trabalho de (PETKOVIC; SOSNICK-PÉREZ; HUANG et al., 2014) o estudo apresenta uma abordagem inovadora para

avaliar e prever os resultados de aprendizagem em equipes de engenharia de software, usando dados de uma classe conjunta em três universidades. A abordagem se concentra na capacidade das equipes de aplicar os melhores processos de engenharia de software e desenvolver produtos. Utiliza medidas objetivas e quantitativas de atividade da equipe, aplicando aprendizado de máquina com o algoritmo Random Forest para prever equipes com baixo desempenho. Os resultados preliminares mostram que essa abordagem é viável, identificando equipes com precisão e destacando fatores importantes. No estudo subsequente (PETKOVIC; SOSNICK-PÉREZ; OKADA et al., 2016), o foco é prever a eficácia da aprendizagem em trabalho em equipe de engenharia de software com base em mais de 100 medidas de atividade de equipe, alcançando uma precisão de cerca de 70%. Isso tem potencial para orientar educadores e gerentes de projetos na intervenção precoce em equipes com dificuldades. Já o trabalho de (FARIA; LIMA, 2019) analisou separadamente os milestones e no último os autores conseguiram 81.81% de acurácia.

No entanto, neste estudo, nosso objetivo é identificar os atributos mais relevantes para análise, utilizando filtros de correlação. Esses filtros são métodos de análise de dados e aprendizado de máquina que avaliam a relação entre atributos em um conjunto de dados (SOUZA; LIMA, 2023). Eles são usados no pré-processamento de dados para reduzir a dimensionalidade, identificando atributos relevantes com base na correlação entre eles. Embora úteis para resolver multicolinearidade e seleção de características importantes (DORNELAS; LIMA, 2023), é importante notar que correlação não implica causalidade. Além disso, em conjuntos de dados extensos, a análise de correlação pode ser computacionalmente intensiva, sugerindo a exploração de outras técnicas de seleção de características. Neste estudo, aplicamos filtros de correlação na busca pelos melhores atributos para a classificação de dados em 5 algoritmos de classificação.

O algoritmo Decision Tree Learner (DTL): Este algoritmo cria uma estrutura de árvore para tomar decisões com base em condições em seus nós. Cada bifurcação da árvore representa uma decisão com base em um atributo, tornando-a fácil de entender e interpretar. É usado para classificação e regressão. Já o Gradient Boosted Learner (GBL): é uma técnica de conjunto de modelos que combina várias árvores de decisão fracas para criar um modelo forte. Ele melhora o desempenho ao ajustar os erros dos modelos anteriores, tornando-o eficaz para problemas de classificação e regressão. O algoritmo Naive Bayes Learner (NBL): Este é um algoritmo probabilístico baseado no

teorema de Bayes. Ele presume independência entre os atributos, o que pode ser uma simplificação excessiva, mas é eficaz para classificação de texto e tarefas de classificação de documentos.

O algoritmo Random Forest Learner (RFL): é um algoritmo de conjunto que cria várias árvores de decisão e combina suas previsões. Isso ajuda a reduzir o overfitting e aumenta a precisão do modelo. É amplamente utilizado em classificação e regressão. Por fim, o algoritmo Support Vector Machine (SVM): é uma técnica de aprendizado supervisionado que encontra um hiperplano de separação ideal entre classes. É eficaz na classificação de dados, especialmente quando há um grande número de atributos. Também pode ser usado para regressão. Cada algoritmo tem suas próprias vantagens e desvantagens, e a escolha depende do tipo de problema e dos dados envolvidos.

Materiais e métodos

A pesquisa tem como objetivo compreender o desempenho das equipes multinacionais de desenvolvimento de software e os fatores que influenciam seu resultado percentual final de aproveitamento. Nesse sentido, uma análise de dados minuciosa e rigorosa é fundamental. Portanto, utilizaremos ferramentas e práticas para processar e calcular diversos resultados a partir da base de dados (BD) que será empregada por meio inteligência artificial. Dado que esta pesquisa está diretamente relacionada à área de TI, ela pode ser classificada como aplicada.

Como a finalidade principal deste trabalho é analisar diferentes equipes de desenvolvimento de software, identificando os fatores que impactam de forma positiva e negativamente em seu desempenho coletivo, bem como demonstrar quais são as melhores formações de equipes de desenvolvimento de software, a natureza desta pesquisa pode ser considerada descritiva. De acordo com (GIL et al., 2002), algumas pesquisas descritivas vão além da simples identificação da existência de relações entre variáveis e buscam determinar a natureza dessas relações. Nesse caso, estamos diante de uma pesquisa descritiva que se aproxima da pesquisa explicativa.

O estudo se baseia em uma série de casos de teste e experimentos com variáveis, fazendo uso da base de dados em formato `.xlsx` em conjunto com a plataforma de análise de dados KNIME Analytics Platform. Durante esse processo, são implemen-

tados algoritmos de aprendizado de máquina para alcançar os objetivos e conclusões desejados. Conforme (GIL et al., 2002) destaca, a pesquisa experimental implica na definição de um objeto de estudo, na seleção das variáveis que podem influenciá-lo e na estipulação de regras para controlar e observar os efeitos que essas variáveis exercem sobre o objeto em questão.

Como materiais usamos um conjunto de dados[1] extraído do repositório de Aprendizado de Máquina da UCI, composto por mais de 100 medidas de atividades de equipe e resultados de aprendizado de máquina obtidos a partir das atividades de 74 equipes de estudantes durante projetos finais de Engenharia de Software em aulas ministradas nas universidades SFSU, Fulda e FAU (PETKOVIC; SOSNICK-PÉREZ; HUANG et al., 2014), desta forma, adotando uma abordagem quantitativa e se caracterizando como um estudo de caso único. Os dados são sequenciais e de séries temporais, relacionados à área de Ciência da Computação e destinados principalmente a tarefas de classificação. As amostras incluem informações numéricas inteiras e reais. Esses dados têm o potencial de serem usados para prever o aprendizado dos estudantes em trabalho em equipe de Engenharia de Software com base na observação de suas atividades em equipe.

Os dados foram coletados ao longo de vários semestres, com o consentimento dos estudantes para fins de pesquisa, garantindo a privacidade dos participantes (PETKOVIC; SOSNICK-PÉREZ; OKADA et al., 2016). As medidas de atividade do estudante foram coletadas de diversas fontes, como cartões de ponto semanais, observações do instrutor e registros de uso de ferramentas de engenharia de software. Os resultados do trabalho em equipe dos estudantes foram avaliados em duas categorias: processo e produto, e classificados em duas classes, A e F, representando equipes que atenderam ou excederam o esperado e aquelas que ficaram aquém.

Os dados estão disponíveis sob a licença Creative Commons Attribution-Non Commercial 4.0 International e foram coletados como parte do Projeto de Avaliação e Previsão de Equipes de Engenharia de Software (SETAP) da Universidade Estadual de São Francisco, financiado em parte pelo subsídio NSF NSF-TUES1140172 (PETKOVIC; SOSNICK-PÉREZ; HUANG et al., 2014). Este conjunto de dados oferece

[1]Data for Software Engineering Teamwork Assessment in Education Setting `https://archive.ics.uci.edu/dataset/393/data+for+software+engineering+teamwork+assessment+in+education+setting`.

uma rica fonte de informações para análise e previsão de desempenho de equipes de engenharia de software em contextos acadêmicos.

Desenvolvimento

O sistema operacional utilizado para a realização deste trabalho e exploração dos resultados foi o Windows 11, por ser mais recente, receber atualizações constantes e ter a compatibilidade necessária com os softwares utilizados, Excel e KNIME. Para análise da base de dados, foi necessário o agrupamento de todos os intervalos de tempo, passando 11 arquivos de extensão `.csv` para 1 arquivo `.xlsx`, adicionando ainda um atributo "Time Interval", referente aos tempos dos processos de Engenharia de Software (ESW).

Na Tabela 1, estão listados os 85 atributos da base de dados, juntamente com a classe (em azul), que serão avaliados quanto à sua relevância para o desempenho das 74 equipes de desenvolvimento de software. Nesse processo, serão identificados e filtrados aqueles que possam impactar negativamente o desempenho, sendo posteriormente excluídos da análise. O arquivo no formato `.xlsx` pode ser lido com o módulo Excel Reader na aplicação de análise de dados KNIME e, em seguida, tratado de acordo com a finalidade do trabalho. Nosso projeto no KNIME Analytics Platform envolveu várias etapas de análise de dados e exploração de atributos, conforme apresentado na Figura 1. Primeiramente, é realizada a Leitura de Dados (Excel Reader): O processo começa com o Reader, responsável por ler a base de dados.

Em seguida, é feita a Normalização (Normalizer): os valores de todas as colunas numéricas são normalizados usando o Normalizer, ajudando a escalar os valores para uma faixa comum e garantindo uma análise precisa. Cálculo de Correlação (Linear Correlation): é usado para calcular a correlação entre duas colunas, considerando os valores ausentes, se houver. Filtro de Correlação (Correlation Filter): entra em ação para filtrar os atributos mais relevantes com base nos cálculos de correlação realizados anteriormente. Validação Cruzada (X-Partitioner): cria um loop de validação cruzada, permitindo avaliar o desempenho do modelo em diferentes conjuntos de dados. Em nosso trabalho, usamos um valor de $X = 10$ loops de validação.

O algoritmo de aprendizado (Learner): representa os algoritmos de aprendiza-

Tabela 1: Apresentação dos 85 atributos avaliados pelos algoritmos de mineração de dados.

Process Atributes and Class	
teamMemberCount	standardDeviationResponsesByStudent
femaleTeamMembersPercent	averageMeetingHoursTotalByStudent
teamLeadGender	standardDeviationMeetingHoursTotalByStudent
teamDistribution	averageMeetingHoursAverageByStudent
teamMemberResponseCount	standardDeviationMeetingHoursAverageByStudent
meetingHoursTotal	averageInPersonMeetingHoursTotalByStudent
meetingHoursAverage	standardDeviationInPersonMeetingHoursTotalByStudent
meetingHoursStandardDeviation	averageInPersonMeetingHoursAverageByStudent
inPersonMeetingHoursTotal	standardDeviationInPersonMeetingHoursAverageByStudent
inPersonMeetingHoursAverage	averageNonCodingDeliverablesHoursTotalByStudent
inPersonMeetingHoursStandardDeviation	standardDeviationNonCodingDeliverablesHoursTotalByStudent
nonCodingDeliverablesHoursTotal	averageNonCodingDeliverablesHoursAverageByStudent
nonCodingDeliverablesHoursAverage	standardDeviationNonCodingDeliverablesHoursAverageByStudent
nonCodingDeliverablesHoursStandardDeviation	averageCodingDeliverablesHoursTotalByStudent
codingDeliverablesHoursTotal	standardDeviationCodingDeliverablesHoursTotalByStudent
codingDeliverablesHoursAverage	averageCodingDeliverablesHoursAverageByStudent
codingDeliverablesHoursStandardDeviation	standardDeviationCodingDeliverablesHoursAverageByStudent
helpHoursTotal	averageHelpHoursTotalByStudent
helpHoursAverage	standardDeviationHelpHoursTotalByStudent
helpHoursStandardDeviation	averageHelpHoursAverageByStudent
averageResponsesByWeek	standardDeviationHelpHoursAverageByStudent
standardDeviationResponsesByWeek	commitCount
averageMeetingHoursTotalByWeek	uniqueCommitMessageCount
standardDeviationMeetingHoursTotalByWeek	uniqueCommitMessagePercent
averageMeetingHoursAverageByWeek	commitMessageLengthTotal
standardDeviationMeetingHoursAverageByWeek	commitMessageLengthAverage
averageInPersonMeetingHoursTotalByWeek	commitMessageLengthStandardDeviation
standardDeviationInPersonMeetingHoursTotalByWeek	averageCommitCountByWeek
averageInPersonMeetingHoursAverageByWeek	standardDeviationCommitCountByWeek
standardDeviationInPersonMeetingHoursAverageByWeek	averageUniqueCommitMessageCountByWeek
averageNonCodingDeliverablesHoursTotalByWeek	standardDeviationUniqueCommitMessageCountByWeek
standardDeviationNonCodingDeliverablesHoursTotalByWeek	averageUniqueCommitMessagePercentByWeek
averageNonCodingDeliverablesHoursAverageByWeek	standardDeviationUniqueCommitMessagePercentByWeek
standardDeviationNonCodingDeliverablesHoursAverageByWeek	averageCommitMessageLengthTotalByWeek
averageCodingDeliverablesHoursTotalByWeek	standardDeviationCommitMessageLengthTotalByWeek
standardDeviationCodingDeliverablesHoursTotalByWeek	averageCommitCountByStudent
averageCodingDeliverablesHoursAverageByWeek	standardDeviationCommitCountByStudent
standardDeviationCodingDeliverablesHoursAverageByWeek	averageUniqueCommitMessageCountByStudent
averageHelpHoursTotalByWeek	issueCount
standardDeviationHelpHoursTotalByWeek	onTimeIssueCount
averageHelpHoursAverageByWeek	lateIssueCount
standardDeviationHelpHoursAverageByWeek	timeInterval
averageResponsesByStudent	SE Process grade

gem usados para treinar o modelo com os dados disponíveis. Usamos 5 modelos de aprendizado de máquina: DTL, GBL, NBL, RFL e SVM. Já o nó de Previsão (Predictor): entra em cena para fazer previsões com base no modelo treinado em cada iteração da validação cruzada. Agregação (X-Aggregator): Os resultados de cada iteração são armazenados pelo X-Aggregator, permitindo uma análise posterior, com $X = 10$ loops no nosso trabalho. Pontuação (Scorer): O Scorer apresenta os resultados, incluindo a precisão do algoritmo em suas previsões. Por fim, o armazenamento (Excel Writer) registra os resultados do Scorer em um arquivo Excel, oferecendo uma organização eficiente das descobertas.

Essas etapas são parte integrante do fluxo de trabalho no KNIME Analytics

Platform e desempenham papéis cruciais na análise e modelagem de dados. Utilizando o projeto KNIME apresentado na Figura 1, foram feitas 10 execuções de 10^2 iterações cada, ajustando o correlation filter (C_F) de 0.1 até 1.0, reduzindo um décimo a cada execução. Esse modelo de análise foi necessário para a identificação de atributos rele-

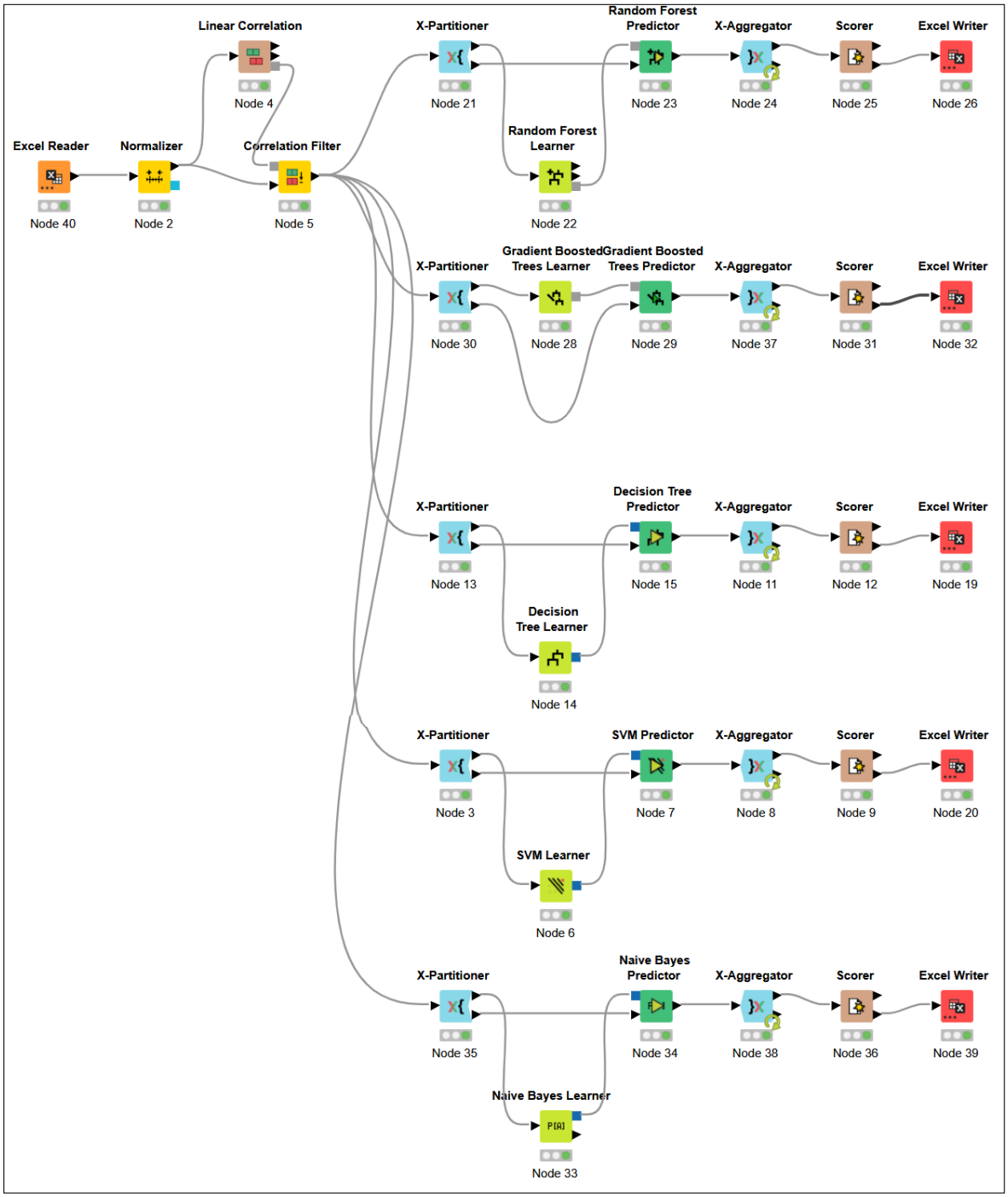

Figura 1: Projeto desenvolvido no KNIME Analytics Platform.

vantes e a sua relação com o aumento de acertos (accuracy). Dessa forma encontramos os seguintes resultados:

Análise e discussão

A Tabela 2 apresenta as acurácias obtidas para diferentes valores de limiar no filtro de correlação, juntamente com cinco algoritmos de classificação de dados. A análise de

Tabela 2: Acurácias para os resultados encontrados pelos algoritmos de classificação de dados.

Correlation Filter	Decision Tree Learner	Gradient Boosteed Learner	Naive Bayes Learner	Random Forest Learner	Support Vector Machine
1.0	0.807062	0.905422	0.701135	0.882724	0.784363
0.9	0.810845	0.901639	0.698613	0.892812	0.773014
0.8	0.843632	0.901639	0.726356	0.880202	0.76797
0.7	0.84111	0.9029	0.732661	0.895334	0.755359
0.6	0.836066	0.904161	0.730139	0.89029	0.760404
0.5	0.836066	0.885246	0.7314	0.899117	0.745271
0.4	0.798235	0.887768	0.728878	0.900378	0.737705
0.3	0.820933	0.887768	0.725095	0.904161	0.716267
0.2	0.790668	0.847415	0.70744	0.851198	0.669609
0.1	0.733922	0.751576	0.706179	0.759142	0.708701

resultados usa o filtro de correlação com vários limiares (C_F) para selecionar atributos relevantes. Com $C_F = 1.0$, todos os 85 atributos são mantidos, e Gradient Boosted Learner e Support Vector Machine lideram com acurácias de 90.54% e 78.44%, respectivamente. Com $C_F = 0.9$, restam 63 atributos, e Gradient Boosted Learner e Random Forest Learner mantêm altas acurácias. Reduzindo para $C_F = 0.5$, com 14 atributos, Gradient Boosted Learner ainda lidera com 90.42% de acurácia. A análise destaca que Gradient Boosted Learner tende a ter o melhor desempenho na maioria dos limiares de filtro de correlação, alcançando uma acurácia mais alta de 90.54%. A Tabela 3 resume as métricas de desempenho dos cinco algoritmos em dois cenários representados por C_F "`F`" (falha) e "`A`" (aprovado).

A Tabela 3 apresenta um resumo das métricas de desempenho para os cinco algoritmos de classificação de dados (DTL, GBT, NBL, RFL e SVM) em dois cenários diferentes, representados pelas colunas C_F (Correlation Filter) com valores "`F`" (failure) e "`A`" (approved). A seguir são realizadas as análises para os algoritmos estudados.

No algoritmo DTL, na classe "`F`" (Filtro de Correlação), o modelo apresenta acurácia de 86.38% e recall de 80.23%, indicando eficácia na detecção de positivos verdadeiros. A precisão é de 79.03%, mantendo equilíbrio entre verdadeiros positivos e falsos positivos. Sensibilidade (TPR) e especificidade (TNR) também são satisfatórias, com 80.23% e 89.43%, respectivamente. O F-measure é de 79.62%, o Kappa de Cohen

Tabela 3: Refinamento dos parâmetros considerando-se os 5 algoritmos de classificação.

Learner Algorithm	C_F	SE Process Grade	TP	FP	TN	FN	Recall	Precision	Sensitivity	Specificity	F-measure	Cohen's kappa	Accuracy
DTL	0.74	F	211	56	474	52	0.802281	0.790262	0.802281	0.89434	0.796226	0.693962	0.863808
		A	474	52	211	56	0.89434	0.901141	0.89434	0.802281	0.897727		
GBT	0.99	F	214	21	509	49	0.813688	0.910638	0.813688	0.960377	0.859438	0.795396	0.911728
		A	509	49	214	21	0.960377	0.912186	0.960377	0.813688	0.935662		
NBL	0.7	F	108	57	473	155	0.410646	0.654545	0.410646	0.892453	0.504673	0.334495	0.732661
		A	473	155	108	57	0.892453	0.753185	0.892453	0.410646	0.816926		
RFL	0.3	F	204	17	513	59	0.775665	0.923077	0.775665	0.967925	0.842975	0.774755	0.904161
		A	513	59	204	17	0.967925	0.896853	0.967925	0.775665	0.931034		
SVM	0.98	F	129	34	496	134	0.490494	0.791411	0.490494	0.935849	0.605634	0.471501	0.788146
		A	496	134	129	34	0.935849	0.787302	0.935849	0.490494	0.855172		

reflete concordância substancial de 69.40%, e a exatidão é de 86.38%. Na classe "A" (Todos os Atributos), o DTL mantém acurácia superior de 86.38%, com alta precisão e recall. No algoritmo GBT, na classe "F", o modelo tem acurácia notável de 91.17%, recall (81.37%), e precisão de 91.02%, com equilíbrio entre verdadeiros positivos e falsos positivos. Sensibilidade (TPR) e especificidade (TNR) atingem bons resultados, com 81.37% e 96.04%. O F-measure é de 85.94%, o Kappa de Cohen reflete concordância de 79.54%, e a exatidão é excelente em 91.17%. Na classe "A", o GBT mantém alto desempenho em acurácia, precisão e recall.

No algoritmo NBL, na classe "F", o modelo tem acurácia razoável de 73.27%, com recall baixo (41.07%), dificuldade na detecção de positivos verdadeiros. A precisão é de 65.45%, com Sensibilidade (TPR) e especificidade (TNR) variando em 41.07% e 89.25%. O F-measure é de 50.47%, o Kappa de Cohen reflete concordância substancial baixa de 33.45%, e a exatidão é razoável em 73.27%. Na classe "A", o NBL mantém desempenho razoável em acurácia, precisão e recall. No algoritmo RFL, na classe "F", o modelo exibe desempenho notável, com acurácia de 90.42%, recall de 77.57%, e alta precisão de 92.31%. Sensibilidade (TPR) e especificidade (TNR) são impressionantes, com 77.57% e 96.79%. O F-measure é de 84.29%, o Kappa de Cohen reflete concordância substancial de 77.48%, e a exatidão é muito boa em 90.42%. Na classe "A", o RFL mantém alto desempenho em acurácia, precisão e recall. No algoritmo SVM, na classe "F", o modelo tem acurácia razoável de 78.81%, com recall baixo (49.05%), indicando dificuldade na detecção de positivos verdadeiros. A precisão é de 79.14%, com Sensibilidade (TPR) e especificidade (TNR) variando em 49.05% e 93.59%. O F-measure é de 60.56%, o Kappa de Cohen reflete concordância substancial de 47.15%, e a exatidão é razoável em 78.81%. Na classe "A", o SVM mantém desempenho razoável em precisão e recall.

A abordagem do algoritmo Gradient Boosted Learner (GBT) com sua alta acu-

rácia de 91.17%, considerando quase todos os 79 atributos da base de dados, é impressionante, porém, pode se tornar computacionalmente custosa para análises e mineração de dados. Por outro lado, o Random Forest Learner (RFL), embora tenha uma acurácia ligeiramente menor de 90.42%, adota uma abordagem mais simplificada, com a consideração de apenas 7 atributos, além da classe SE Process Grade. Essa escolha torna o RFL uma opção mais interessante do ponto de vista computacional. A seleção desses atributos, como "teamMemberCount", "femaleTeamMemberPercent", "inPersonMeetingHoursStandardDeviation", "averageNonCodingDeliverablesHoursTotalByWeek", "standardDeviationResponsesByStudent", "averageCodingDeliverablesHoursTotalByStudent" e "commitMessageLengthStandardDeviation", revela sua relevância para a análise e gestão de equipes de desenvolvimento de software no contexto de trabalho remoto, fornecendo insights valiosos com eficiência computacional. Portanto, a escolha entre GBT e RFL deve levar em consideração não apenas a acurácia, mas também a praticidade e eficiência em relação ao uso de recursos computacionais, dependendo das necessidades específicas da análise e dos objetivos de negócios.

Conclusões

Neste estudo, conduzimos uma análise abrangente dos resultados obtidos por diferentes algoritmos de classificação de dados aplicados a um conjunto de dados relacionados à gestão de equipes de desenvolvimento de software em regime de trabalho remoto. Observamos que o algoritmo GBT demonstrou um desempenho notável, alcançando a maior acurácia de 91.17% ao considerar a maioria dos atributos disponíveis, embora essa abordagem possa ser computacionalmente intensiva. Por outro lado, o RFL, com uma acurácia ligeiramente menor de 90.42%, adotou uma estratégia mais simplificada, considerando apenas um conjunto reduzido de atributos, tornando-se uma alternativa mais eficiente do ponto de vista computacional. A seleção criteriosa desses atributos revelou sua relevância na análise e gestão de equipes remotas de desenvolvimento de software. Os atributos-chave identificados, como composição da equipe e horas de reunião, são relevantes na gestão de equipes de desenvolvimento de software que operam remotamente. Portanto, as conclusões destacam a importância de equilibrar desempenho preditivo com eficiência computacional ao escolher o algoritmo de classificação

mais adequado para tarefas de análise de dados similares, garantindo que os recursos sejam alocados de maneira eficaz de acordo com os objetivos da pesquisa e as necessidades práticas. Em futuras pesquisas, planejamos classificar equipes de desenvolvimento de software em contextos de fábrica de software.

Referências

DORNELAS, R. S.; LIMA, D. A. Correlation Filters in Machine Learning Algorithms to Select Demographic and Individual Features for Autism Spectrum Disorder Diagnosis. **Journal of Data Science and Intelligent Systems**, 2023.

FARIA, D. M. C.; LIMA, D. A. Mineracao de dados para a avaliacao dos atributos que afetam o processo de desenvolvimento de sistemas. **Trabalho de Conclusao de Curso em Analise e Desenvolvimento de Sistemas**, 2019.

GIL, A. C. et al. **Como elaborar projetos de pesquisa**. [S.l.]: Atlas São Paulo, 2002. v. 4.

PERDIGAO, C.; SEDA, V. B. et al. Percepção dos consumidores sobre o home office no período da pandemia. FGV IBRE, 2022.

PETKOVIC, D.; SOSNICK-PÉREZ, M.; HUANG, S. et al. Setap: Software engineering teamwork assessment and prediction using machine learning. In: IEEE. 2014 IEEE frontiers in education conference (FIE) proceedings. [S.l.: s.n.], 2014. P. 1–8.

PETKOVIC, D.; SOSNICK-PÉREZ, M.; OKADA, K. et al. Using the random forest classifier to assess and predict student learning of software engineering teamwork. In: IEEE. 2016 IEEE frontiers in education conference (FIE). [S.l.: s.n.], 2016. P. 1–7.

SOUZA, V. S.; LIMA, D. A. Identifying Risk Factors for Heart Failure: A Case Study Employing Data Mining Algorithms. **Journal of Data Science and Intelligent Systems**, 2023.

TASCHETTO, M.; FROEHLICH, C. Teletrabalho sob a perspectiva dos profissionais de recursos humanos do Vale do Sinos e Paranhana no Rio Grande do Sul. **Revista de Carreiras e Pessoas**, v. 9, n. 3, 2019.