UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DOGLAS ANDRÉ FINCO

COMBINANDO *PLANNING POKER* E APRENDIZADO DE MÁQUINA PARA ESTIMAR ESFORÇO DE SOFTWARE

CURITIBA

2022

DOGLAS ANDRÉ FINCO

COMBINANDO PLANNING POKER E APRENDIZADO DE MÁQUINA PARA ESTIMAR ESFORÇO DE SOFTWARE

Combining Planning Poker and machine learning to estimate software effort

Dissertação apresentada como requisito para obtenção do título de Mestre em Computação Aplicada, do Departamento de Informática da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Prof. Dr. Laudelino Cordeiro Bastos

Coorientador: Prof. Dr. Adolfo Gustavo Serra Seca Neto

CURITIBA

2022

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Curitiba

DOGLAS ANDRE FINCO

COMBINANDO PLANNING POKER E APRENDIZADO DE MÁQUINA PARA ESTIMAR ESFORÇO DE SOFTWARE

Trabalho de pesquisa de mestrado apresentado como requisito para obtenção do título de Mestre Em Computação Aplicada da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Engenharia De Sistemas Computacionais.

Data de aprovação: 08 de Dezembro de 2021

Prof Laudelino Cordeiro Bastos, Doutorado - Universidade Tecnológica Federal do Paraná

Prof Adolfo Gustavo Serra Seca Neto, Doutorado - Universidade Tecnológica Federal do Paraná

Prof Giancarlo Dondoni Salton, Doutorado - Universidade do Oeste de Santa Catarina (Unoesc)

Prof.a Maria Claudia Figueiredo Pereira Emer, Doutorado - Universidade Tecnológica Federal do Paraná

Documento gerado pelo Sistema Acadêmico da UTFPR a partir dos dados da Ata de Defesa em 08/12/2021.

AGRADECIMENTOS

Certamente estes parágrafos não irão atender a todos (as) que fizeram parte dessa importante fase de minha vida. Portanto, desde já peço desculpas aos que não estão presentes entre essas palavras, mas elas podem estar certas que fazem parte do meu pensamento e de minha gratidão.

Agradeço primeiramente a Deus pela saúde e por me guiar com sabedoria nesta trajetória, especialmente nos momentos de dificuldade.

À minha família, pais Gilmar e Elenice, irmão Gladson e minha noiva Amanda por me apoiarem no desafio. Por me ajudarem nas idas/vindas até a rodoviária, ajuda nos custos e acima de tudo, apoio moral para chegar até aqui.

Agradeço em especial a meu orientador Prof. Dr. Laudelino Cordeiro Bastos e coorientador Prof. Dr. Adolfo Gustavo Serra Seca Neto por me guiarem com muita dedicação, sabedoria e compreensão em todos os momentos. Além deles, aos professores que compuseram a banca avaliadora, Prof. Dr. Maria Cláudia Emer e Prof. Dr. Giancarlo Dondoni Salton pelas considerações e sugestões pertinentes para a melhoria do trabalho.

Também, aos colegas que conheci e pela parceria criada durante este período, são amigos que levo para a vida. Enfim, a todos que por algum motivo contribuíram para a realização desta pesquisa.

Embora estimar seja muito mais arte do que ciência, não precisa ser conduzida de maneira aleatória. [...]. Por serem a base para todas as outras ações do planejamento do projeto, e pelo fato de o planejamento de projeto fornecer a direção para uma engenharia de software bem-sucedida, seria uma péssima ideia iniciar sem as estimativas (PRESSMAN, 2011).

RESUMO

Estimar esforço de software é crítico às organizações, pois subestimativa ou superestimativa podem resultar em falhas nos projetos. O *Planning Poker* é uma das práticas mais utilizadas para definir estimativas de esforço. A estimativa ocorre com base na experiência dos integrantes, mediante reunião envolvendo todos os membros da equipe. Porém, as informações geradas neste debate não são guardadas devido à informalidade da prática e como esse conhecimento se perde, não há como aproveitá-lo em estimativas futuras. A aplicação de técnicas de aprendizado de máquina (AM) nas estimativas de esforço cresceu nos últimos anos, usadas complementarmente ou alternativamente a outras abordagens. Estudos apontam que a utilização de práticas combinadas proporciona maior assertividade em relação a técnicas individuais. Assim sendo, este estudo objetivou descrever a combinação do *Planning Poker* com AM, abordagem criada e nomeada *ML Planning Poker,* e avaliar se a proposta interfere no processo de estimativas de esforço. Realizamos pesquisa bibliográfica, um mapeamento sistemático e uma *survey* fortalecendo as bases do estudo. Fundamentados nas descobertas, descrevemos as etapas da *ML Planning Poker* e desenvolvemos uma ferramenta servindo de meio interativo com as equipes no processo de estimativas, tal que embutido a ela criamos um modelo de AM. Na sequência, documentamos a proposta e, avaliamos com estudantes de graduação e profissionais de TI. A avaliação com estudantes resultou em assertividade idêntica nas tarefas estimadas usando o *Planning Poker* original e a proposta. Porém, no que se refere as tarefas com estimativa incorreta, percebemos que a *ML Planning Poker* teve melhor resultado, já que, 57,1% das tarefas teve diferença de no máximo 1 hora entre tempo estimado e realizado em comparação a 39,2% do *Planning Poker* original. Além disso, dos estudantes participantes, 81,2% concordam que a proposta contribui com o processo de estimativas. Os profissionais de TI perceberam benefícios da proposta e defendem que o AM proporciona um subsídio aos integrantes. Reforçaram também o problema do esquecimento de tarefas muito antigas dificultando a estimativa atual, sendo que a *ML Planning Poker* auxilia, pois traz tarefas semelhantes anteriores. Mesmo que existam questões a serem melhoradas, como a acurácia do modelo e a usabilidade da ferramenta, percepções direcionam para benefícios da *ML Planning Poker* nas tarefas realizadas há muito tempo ou que os membros não possuem experiência, trazendo maior segurança ao participante na definição de sua estimativa. A *ML Planning Poker* apresenta potencial pois o fator humano do *Planning Poker* continua sendo considerado enquanto o AM apoia a tomada de decisão, permitindo melhorar o processo de estimativas.

Palavras-chave: Estimativa de esforço de software. Planning Poker. Aprendizado de Máquina. Combinação de técnicas.

ABSTRACT

Estimating software effort is a critical factor to organizations, as underestimation or overestimation can result in project failures. Planning Poker is one of the most used practices to define effort estimates. The estimate is based on the experience of the members, through a meeting involving all team members. However, the information generated in this debate is not kept due to the informality of the practice and how that knowledge is lost, there is no way to use it in future estimates. The application of machine learning (ML) techniques in effort estimates has grown in recent years, used in addition to or alternatively to other approaches. Studies indicate that the use of combined practices provides greater assertiveness in relation to individual techniques. Therefore, this study aimed to describe the combination of Planning Poker with AM, an approach created and named ML Planning Poker, and assess whether the proposal interferes with the effort estimation process. We carry out bibliographic research, a systematic mapping and a survey strengthening the study bases. Based on the findings, we describe the steps of ML Planning Poker and we developed a tool that supports and an interactive medium with the teams in the estimation process, such that built into it we create an ML model. After that, we document the proposal and, we evaluate with undergraduate students and IT professionals. The evaluation with students resulted in identical assertiveness in the tasks estimated using the original Planning Poker and the proposal. Although, regarding tasks with incorrect estimation, we noticed that ML Planning Poker had a better result, given that, 57.1% of the tasks had a difference of at most 1 hour between estimated and performed time compared to 39.2% of the original Planning Poker. Furthermore, of the participating students, 81.2% agree that the proposal contributes to the estimation process. IT professionals realized benefits of the proposal and defended that ML provides a subsidy to members. They also reinforced the problem of forgetting very old tasks, making the current estimate difficult, and ML Planning Poker helps, as it brings similar tasks from the previous ones. Even though there are issues to be improved, such as the accuracy of the model and the usability of the tool, insights point to the benefits of ML Planning Poker in tasks that have been performed for a long time or that members have no experience, bringing greater security to the participant in the definition of his estimate. ML Planning Poker has potential because the human factor present in Planning Poker continues to be considered while ML supports decision making, allowing to improve the estimation process.

Keywords: Software effort estimation. Planning Poker. Machine Leaning. Combination of techniques.

LISTA DE ILUSTRAÇÕES

FIGURAS

GRÁFICOS

QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS

LISTA DE SIGLAS

AM	Aprendizado de Máquina
CRISP-DM	*Cross Industry Standart Process for Data Mining*
GBM	*Gradient Boosting Machine*
IA	Inteligência Artificial
k-NN	*K-Nearest Neighbors*
RF	Requisito Funcional
RNF	Requisito Não-Funcional
TI	Tecnologia da Informação
UC	*Use Case*
UML	*Unified Model Language*
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1 INTRODUÇÃO

A estimativa de esforço de desenvolvimento de software é o processo de prever o esforço necessário para desenvolver um sistema de software (WEN *et al.*, 2012; TISSOT *et al.*, 2015). Ziauddin e Zia (2012) afirmam que estimar esforço em projetos de software é um ponto crítico para toda organização. A subestimativa pode resultar em pressão de tempo, comprometendo o desenvolvimento funcional. Da mesma forma, a superestimativa pode culminar em orçamentos não competitivos (TRONTO *et al.*, 2008). De acordo com Wen *et al.* (2012) estimar o esforço com precisão no estágio inicial do ciclo de vida do software é crucial na eficácia do gerenciamento de projeto.

Similarmente, Park e Baek (2008) defendem que estimativa imprecisa de esforço de desenvolvimento reduz a proficiência do projeto, desperdiça o orçamento da empresa e pode resultar em falha. Tais impactos despertaram diversas pesquisas sobre o tema nos últimos 30 anos. Porém, apesar dos estudos, os fatores que influenciam o esforço de desenvolvimento e as relações entre eles ainda não foram compreendidos e definidos corretamente.

Bloch *et al.* (2012) conduziram uma pesquisa em colaboração com a Universidade de Oxford e sugeriram que metade dos grandes projetos de TI - definidos como aqueles com preços iniciais superiores a USS 15 milhões - estouram maciçamente seus orçamentos. Em média, grandes projetos de TI ficam 45% acima do orçamento e 7% acima do tempo, ao mesmo tempo que fornecem 56% menos valor agregado do que o previsto.

Mesmo com o surgimento e aplicação de técnicas de estimativas, o percentual de falhas ainda é grande, indicando que evolução é necessária. O tema foi abordado por Eman e Koru (2008) num estudo entre os anos 2005 e 2007 considerando falhas gerais (projetos cancelados ou entregues com desempenho malsucedido), os números indicaram que 26% a 34% dos projetos de Tecnologia da Informação (TI) ainda falham. Os autores ressaltam também, que apesar de muitos anos de pesquisa, a habilidade de estimar continua sendo um desafio aos projetos de TI, pois os profissionais não usam as melhores ferramentas e técnicas de estimativa ou porque as melhores ferramentas e técnicas exigem melhorias antes de serem usadas efetivamente.

De acordo com Tronto *et al*. (2008) as técnicas/métodos de estimativas de esforço de software podem ser classificadas em três categorias: opinião especializada, modelos algorítmicos e aprendizado de máquina. No contexto da opinião especializada a prática *Planning Poker* é uma das mais difundidas.

No *Planning Poker* todos os membros da equipe discutem sobre a tarefa de software que está sendo estimada, usando um baralho de cartas no apoio a definição da estimativa. Cada participante escolhe uma carta que se refere ao valor de sua estimativa mantendo-a virada para baixo e, quando todos definiram suas cartas, os valores são revelados simultaneamente. Se os resultados divergirem, os participantes com maior e menor valores justificam suas escolhas e, posteriormente, as cartas retornam ao baralho para a repetição das etapas. O processo ocorre até ser atingida uma estimativa razoável ou consenso, mediado por um moderador. Este processo se repete para todas as tarefas da *sprint* atual.

Na Revisão Sistemática de Dantas *et al.* (2018), *Planning Poker* é a prática mais citada, sendo considerada eficaz quando a equipe tem níveis semelhantes de experiência ao trabalhar em novos projetos. Na *survey* de Usmann *et al.* (2015) com 60 profissionais ágeis de 16 países foi descoberto que o *Planning Poker* é utilizado por 63% dos participantes.

Porém, segundo Tissot *et al.* (2015) as informações geradas pelo debate no *Planning Poker* não são guardadas devido à informalidade do método e como esse conhecimento se perde, não há como aproveitá-lo em estimativas futuras. Ressaltam também, que sua acurácia depende da habilidade do estimador e sua capacidade em estimar a produtividade da equipe e, como ator principal do *Planning Poker*, o ser humano pode ter sua capacidade de decisão afetada por características ambientais ou psicológicas.

Os resultados de outros estudos de julgamento humano indicam que as pessoas ficam excessivamente otimistas ao prever o próprio desempenho, elas têm problema em separar “desejo” e “realismo” (JORGENSEN, 2004). Por sua vez, Mahnic e Hovelja (2012) assumem que a discussão em grupo ocorrida no *Planning Poker* auxilia na identificação de atividades que individualmente poderiam ser ignoradas, proporcionando estimativas mais precisas e reduzindo o excesso de otimismo que geralmente ocorre em métodos baseados em julgamentos individuais.

A utilização do *Planning Poker* permite a troca de conhecimento entre estimadores e diminui o excesso de otimismo que geralmente ocorre em

julgamentos individuais. No entanto, o fato de as informações não serem guardadas, a subjetividade, entre outros aspectos humanos e a falta de experiência com tarefas semelhantes pode resultar em falhas.

Visando superar problemas relacionados às técnicas que considerem apenas a experiência, busca-se apoio em dados históricos. Na revisão sistemática realizada por Nguyen-Cong e Tran-Cao (2013) foi defendida a necessidade de estudos que considerem informações de projetos anteriores na realização de estimativas atuais, permitindo a utilização de dados históricos dos projetos e servindo de comparação com o projeto que está sendo estimado.

Da mesma forma, Jorgensen (2004) defende o uso de dados documentados da empresa, afirmando que os estimadores têm a oportunidade de aplicar uma estratégia mais analítica e menos propensa a vieses humanos e situacionais. Segundo ele, muitas vezes é utilizada a estratégia de estimativa de "representatividade", ou seja, as pessoas usam o esforço real da tarefa lembrada mais similar (mais representativa) como ponto de partida, sem considerar o esforço de outras tarefas similares.

De acordo com Benala *et al.* (2012) mesmo que as teorias de estimativas tradicionais tenham evoluído nas últimas décadas, ainda são consideradas insatisfatórias. Além disso, o desenvolvimento de software é em grande parte esforço humano; assim, as reações humanas em situações específicas e seus sentimentos devem ser considerados. Neste sentido, visando superar as dificuldades citadas, percebemos o aumento da utilização de Inteligência Artificial (IA), através de técnicas de Aprendizado de Máquina (AM) na realização de estimativas de esforço de software.

Dantas *et al.* (2018) afirma que desde 2014 a comunidade científica vem sendo bastante ativa na área de estimativa de esforço no desenvolvimento ágil de software, porém, o tema continua sendo desafiador e objeto de mais estudos, dada a dificuldade em encontrar soluções precisas ao problema. O autor reforça em sua revisão sistemática que uma quantidade significativa de novos trabalhos tem usado técnicas de IA ou AM para apoiar a estimativa de esforço no desenvolvimento ágil, contribuindo com a precisão do processo.

Na Revisão Sistemática de Wen *et al.* (2012), abordando o período entre 1990-2010, foram identificadas técnicas, a precisão de estimativa e os contextos de aplicação das técnicas de IA. Descobriu-se que normalmente as técnicas são

usadas individualmente ou combinadas a outras técnicas, sem a influência do fator humano no processo. Entretanto, com o aumento do uso de métodos ágeis que apresentam em seus valores, “indivíduos e interações mais que processos e ferramentas” (MANIFESTO ÁGIL, 2001), cresce a valorização do indivíduo no processo, pois são os responsáveis pela execução das tarefas. Deste modo, acreditamos que usar apenas IA sem a opinião dos envolvidos pode não ser a melhor escolha, visto que, aspectos de vivência particular a cada membro podem ser ignorados e também, devido ao cenário da empresa que se altera ao longo do tempo em termos de funcionários, experiência, tecnologias de apoio, entre outros.

A comunidade científica propôs diversas técnicas e práticas para alcançar alta acurácia na previsão de esforço. Porém, apesar do grande número de estudos, não há consenso sobre o melhor método único, despertando possibilidades de combinação de técnicas para estimar esforço de software. Na Revisão Sistemática realizada por Idri *et al.* (2016) é defendida a necessidade de superar as fraquezas das técnicas únicas originando o agrupamento das mesmas. Além disso, a revisão concluiu que técnicas em conjunto geralmente são mais precisas que as individuais.

Na survey de Usmann *et al.* (2015) com intuito de explorar o estado da prática nas estimativas ágeis de software, o *Planning Poker* foi selecionado por 38 (63%) praticantes ágeis, dos quais é usado individualmente por 18 (47%) e em combinação com outras técnicas por 20 (53%). As combinações ocorreram juntamente com: Estimativa por analogia (6 praticantes); Estimativa por analogia e opinião de especialistas (5 praticantes); Estimativa por analogia e Delphi (2 praticantes); Estimativa por analogia e pontos de função COSMIC (1 praticante); Estimativa por analogia, opinião de especialistas, pontos de casos de uso e SLIM (1 praticante); Opinião especializada (3 praticantes); e Método de pontos de caso de uso (2 profissionais). Estes dados reforçam que combinar *Planning Poker* com outros métodos vem sendo explorado por seus utilizadores.

Perante a consolidação da possibilidade de combinar técnicas para estimar esforço, nossa hipótese é de que combinando *Planning Poker* com AM pode ser uma alternativa interessante trazendo resultados satisfatórios. O *Planning Poker* valoriza o fator humano e o AM atua no apoio à tomada da decisão trazendo dados históricos. Diante do cenário discutido, definimos a seguinte questão de pesquisa:

- Como realizar uma combinação do *Planning Poker* com AM e avaliar se a combinação interfere no processo de estimativas de esforço de software?

Assim, este estudo objetivou descrever a combinação do *Planning Poker* com AM, nomeada *ML Planning Poker* e avaliar se a proposta interfere no processo de estimativas de esforço de software. Neste contexto, as estimativas são guardadas, e a aplicação do AM permite analisar situações anteriores, sem ignorar o fator humano no processo, tal que o *Planning Poker* continua sendo realizado. Indo ao encontro e apoiado no objetivo geral, foram definidos os seguintes objetivos específicos:

- Avaliar se os resultados trazidos através da ferramenta apoiam na definição da estimativa de esforço de software;
- Testar se o desempenho da *ML Planning Poker* é positivo quanto à utilização do *Planning Poker* original;
- Identificar se os envolvidos ao processo de estimativas aprovam a utilização da *ML Planning Poker*.

1.1 Organização do documento

O presente documento encontra-se organizado da seguinte forma: o Capítulo 2 apresenta a fundamentação teórica, o Capítulo 3 aborda os trabalhos relacionados, o Capítulo 4 traz os materiais e métodos, no Capítulo 5 temos os resultados e discussão, no Capítulo 6 as conclusões e na sequência são apresentadas as referências. Criamos alguns apêndices que também são encontrados: Apêndice A – Questionários de Avaliação, Apêndice B - *Features* usadas no modelo de ML, Apêndice C - Tarefas desenvolvidas na avaliação antes/depois e Apêndice D – Termo de Consentimento Livre e Esclarecido.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo exibe o referencial teórico e foi dividido em cinco seções: Estimativas de esforço de software, *Planning Poker*, Inteligência Artificial, Aprendizado de Máquina nas estimativas de esforço de software e Base de conhecimento de software.

2.1 Estimativas de esforço de software

A estimativa do esforço é essencial para o sucesso dos projetos de software, pois a exigência por precisão é crucial visando atender ao plano dentro do orçamento definido. A precisão de orçamento é muito importante para as empresas de software, a fim de manter sua competitividade no mercado (ABDELLATIF, 2018).

O processo de estimar as atividades de software consiste em definir o número de períodos necessários para realizar as atividades específicas com os recursos estimados. As entradas para as estimativas de duração das tarefas originam-se da pessoa ou grupo na equipe do projeto que está mais familiarizada com a natureza do trabalho na atividade específica. De acordo com o desenvolvimento do projeto, dados mais detalhados e precisos tornam-se disponíveis e a precisão de estimativas de duração melhora (PMBOK, 2017).

Neste contexto Pressman (2011, p. 605) defende que,

> Embora estimar seja muito mais arte do que ciência, não precisa ser conduzida de maneira aleatória. Existem técnicas úteis para estimar tempo e esforço. As métricas de projeto e processo podem proporcionar perspectivas históricas e valiosas informações para gerar estimativas quantitativas. A experiência (de todos os envolvidos) pode ajudar imensamente à medida que as estimativas são desenvolvidas e revisadas. Por serem a base para todas as outras ações do planejamento de projeto, e pelo fato de o planejamento de projeto fornecer a direção para uma engenharia de software bem-sucedida, seria uma péssima ideia iniciar sem as estimativas.

De acordo com Nassif *et al.* (2013) projetos de software falham devido a imprecisão da estimativa de software e incompreensão ou incompletude dos requisitos. Isso motivou pesquisadores a investigar o tema com intuito de melhorar a avaliação do tamanho e esforço do software. À medida que a estimativa se torna

crucial para prevenir ou reduzir as falhas do projeto, ocorrem nos estágios iniciais do ciclo de vida do software torna-se imperativa e quanto mais cedo realizadas melhor o gerenciamento do projeto.

As estimativas de custo e esforço de software nunca serão uma ciência exata. Muitas variáveis (fatores humanos, técnicos, ambientais e políticos) podem afetar custo e esforço necessário para o desenvolvimento de software. No entanto, podem ser transformadas de algo sobrenatural para uma série de etapas sistemáticas permitindo resultados com risco aceitável (PRESSMAN, 2011).

Tronto *et al.* (2008) informa que, atualmente as técnicas ou práticas de estimativas de esforço de software são classificadas em três categorias: opinião especializada, modelos algorítmicos e aprendizado de máquina.

A opinião especializada é realizada de forma implícita, apoiada apenas na experiência individual dos membros baseando-se em projetos anteriores. Dentre estas práticas, podemos citar: o Método Delphi e o *Planning Poker*. Os modelos algorítmicos têm como principal responsável pelo esforço, o tamanho do software (por exemplo: quantidade de pontos de função, linha de código fonte) e, dentre estas técnicas, aparecem: COCOMO, Análise de Pontos de Função e SLIM. Por fim, aparecem as técnicas de aprendizado de máquina, que nas últimas décadas têm sido usadas como complemento ou alternativa para as categorias anteriores. Exemplos desta abordagem incluem: Raciocínio Baseado em Casos, Naive Bayes, Árvores de Decisão e Redes Neurais Artificiais.

As técnicas e práticas citadas possuem particularidades e são utilizadas de acordo com as especificidades dos projetos e preferência das empresas. Nos últimos anos destaca-se a aplicação de técnicas de aprendizado de máquina nas estimativas de esforço, muitos são os trabalhos que exploram o tema.

No que se refere às estimativas no desenvolvimento ágil, Ziauddin e Zia (2012) afirmam que a aplicação de técnicas ou práticas em tais tipos de projetos é muito difícil, mas é uma tarefa importante. As técnicas clássicas de estimativa precisam de requisitos bem definidos e as metodologias ágeis nem sempre suportam esse comportamento, pois elas veem alterações como desafio importante. Tudo isso faz da estimativa no desenvolvimento ágil de software uma tarefa desafiadora.

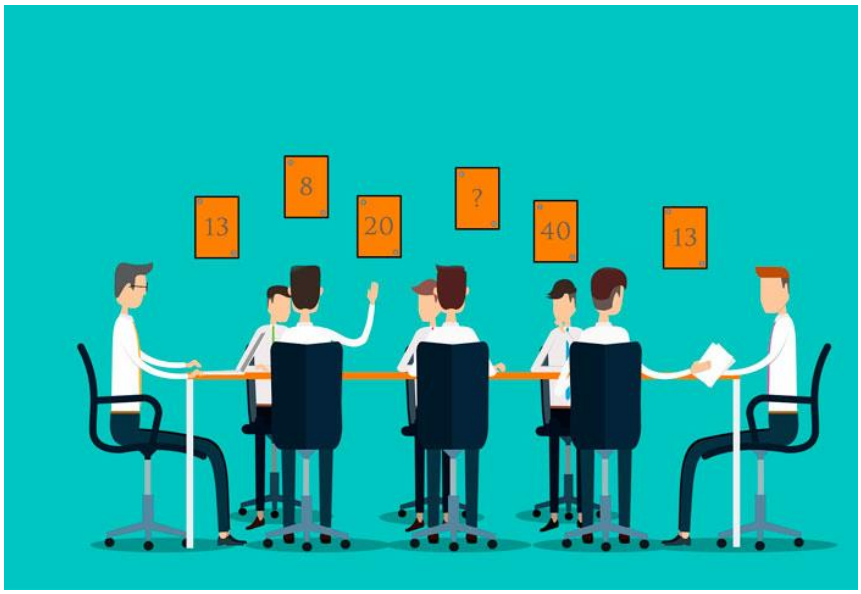
2.2 *Planning Poker*

O termo *Planning Poker* foi introduzido por Grenning (2002) no contexto do desenvolvimento ágil de software. De acordo com Cohn (2005) o *Planning Poker* consiste da combinação de especialistas e separação das partes envolvidas em uma abordagem “agradável” de estimar, apresentando resultados confiáveis rapidamente. Esta forma de estimar considera a experiência dos componentes da equipe, ou seja, decisões baseadas no conhecimento empírico dos membros. A seguir é descrita a sequência de etapas do *Planning Poker* apresentadas por Cohn (2005):

1. Cada membro recebe um baralho de cartas. Cada carta tem escrito sobre ela uma estimativa válida. As estimativas válidas possuem relação com sequência de Fibonacci e geralmente contém os valores 0, 1/2, 1, 2, 3, 5, 8, 13, 20, 40, e 100;
2. Para cada tarefa, um moderador que geralmente é um representante do cliente lê a descrição da tarefa. O moderador responde todas as perguntas que os membros da equipe têm relacionado à tarefa;
3. Após as perguntas serem respondidas, cada estimador seleciona de maneira privada uma das cartas que representa sua estimativa. A carta escolhida é baseada no conhecimento do estimador. As cartas não são mostradas até que todos os membros tenham feito suas escolhas;
4. A partir do momento em que todo membro realizou sua escolha, as cartas são viradas simultaneamente para que todos possam ver cada estimativa;
5. Nesta etapa, os resultados podem divergir significativamente. Se isso ocorrer, os estimadores que escolheram os valores mais alto e mais baixo explicam os fatores analisados e os motivos para a escolha daquela estimativa;
6. O grupo discute sobre a tarefa e suas estimativas por alguns minutos. Após a discussão cada membro seleciona novamente uma carta. Mais uma vez a carta é mantida em sigilo até que todos os membros tenham feito suas escolhas. Isso se repete até que haja um consenso.

A Figura 1 apresenta o cenário de realização do *Planning Poker*.

Figura 1 – Reunião do *Planning Poker*



Fonte: Brasileiro (2017)

Em diversos casos os valores convergem na segunda rodada e raramente ocorrem mais de três rodadas. O objetivo é convergir para uma estimativa única que seja usada na tarefa. Não é necessário que todas as cartas sejam exatamente iguais, pois o objetivo não é a precisão absoluta, mas sim razoabilidade. Um ponto positivo da prática é que à medida que os membros justificam suas escolhas ocorre a disseminação do conhecimento entre as pessoas envolvidas (COHN, 2005).

De acordo com Borte *et al.* (2012) o que torna o *Planning Poker* interessante é: a) fato de permitir que todos os membros da equipe, independentemente da posição ou experiência participem; b) requer explicações dos membros que fornecem maior e menor estimativa; e c) utilizam as cartas de jogo como artefato para realização de estimativas. O método também pode ser entendido como uma forma de organizar o trabalho em grupos de especialistas de diferentes áreas para resolver problemas de alta complexidade na indústria de software.

2.3 Inteligência Artificial

Russel e Norvig (2013) contextualizam que durante milhares de anos procurou-se entender como pensamos, isto é, como uma mera quantidade de matéria pode perceber, compreender, prever e manipular um mundo muito maior e mais complexo que ela própria. O campo da IA vai além, não apenas procura compreender, mas também construir entidades inteligentes.

No início, a IA era vista como uma área teórica com aplicações apenas em pequenos problemas curiosos e desafiadores, mas de pouco valor prático. A partir da década de 1970 o cenário mudou e houve maior disseminação do uso de técnicas de computação baseadas em IA para a solução de problemas reais. Nas últimas décadas, com a crescente complexidade dos problemas tratados computacionalmente e do volume de dados gerados, tornou-se clara a necessidade de ferramentas computacionais mais sofisticadas, que fossem mais autônomas reduzindo a necessidade de intervenção humana e dependência de especialistas. Para isso, essas técnicas deveriam ser capazes de criar por si só a partir da experiência passada, uma hipótese (ou função) capaz de resolver o problema que se deseja tratar (FACELI *et al.*, 2011).

Faceli *et al.* (2011) complementa que a esse processo de indução de uma hipótese (ou aproximação de função) a partir da experiência passada dá-se o nome de Aprendizado de Máquina (AM). Em AM os computadores são programados para aprender com a experiência passada, assim, algoritmos aprendem a induzir uma função ou hipótese capaz de resolver um problema a partir de dados que representam instâncias do problema a ser resolvido. A Seção a seguir aprofunda explanações em volta do tema.

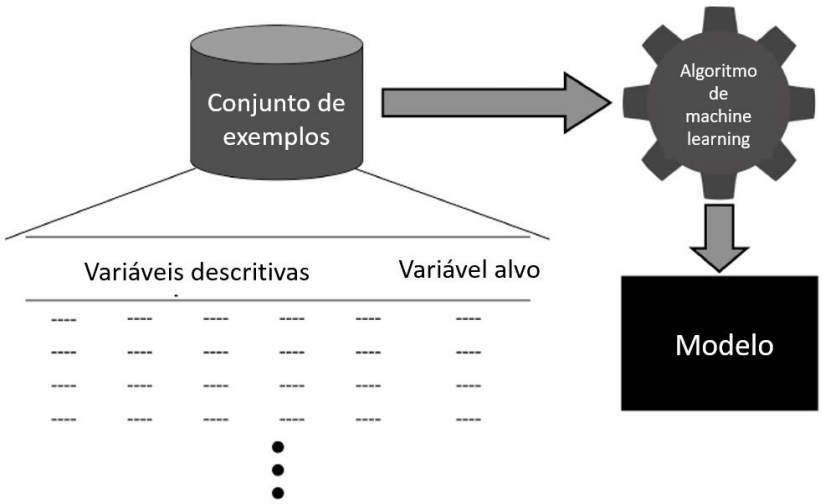
2.3.1 Aprendizado de Máquina

Segundo Shalev-Shwartz e Bem-David (2014) o AM é um subcampo da ciência da computação, pois objetiva programar máquinas para que aprendam. É visto como um ramo da IA, afinal, a capacidade de transformar a experiência em conhecimento ou detectar padrões em dados sensoriais complexos é uma pedra angular da inteligência humana (e animal). Porém, ao contrário da IA tradicional, o AM não está tentando construir imitação automatizada de comportamento inteligente, mas usar os pontos fortes e habilidades especiais dos computadores para complementar a inteligência humana, executando tarefas que vão além das capacidades humanas. Por exemplo, a capacidade de processar grandes bancos de dados permite que programas de AM detectem padrões que estão fora do escopo da percepção humana.

De acordo com Alpaydin (2020) o AM está programando computadores para otimizar um critério de desempenho usando dados de exemplo ou experiências anteriores. Neste contexto temos um modelo definido e seus parâmetros, sendo que o aprendizado é a execução de um programa de computador para otimizar os parâmetros do modelo usando os dados de treinamento ou a experiência passada.

Kelleher *et al.* (2015) define o AM como um processo automatizado que extrai padrões de dados. Para construir os modelos usados em aplicativos preditivos de análise de dados, usa-se aprendizado de máquina supervisionado. O autor complementa que as técnicas supervisionadas de aprendizado de máquina induzem automaticamente um modelo do relacionamento entre um conjunto de variáveis descritivas e uma variável alvo com base em um conjunto de exemplos históricos ou instâncias. Podemos então usar esse modelo para fazer previsões para novas instâncias. Esses dois passos são mostrados nas Figuras 2 e 3.

Figura 2 - Ensinando um modelo a partir de um conjunto de instâncias históricas



Fonte: Adaptado de Keheller *et al.* (2015)

Figura 3 - Usando um modelo para fazer previsões

Fonte: Fonte: Adaptado de Keheller *et al.* (2015)

AM é uma das áreas da computação em expansão nos últimos anos. Diferentes algoritmos de AM, formas de utilizá-los e adaptações são continuamente propostas. Além disso, surgem a todo instante variações nas características dos

problemas reais a serem tratados. Existem inúmeras aplicações bem-sucedidas na solução de problemas reais, entre elas: predição de taxas de cura de pacientes, condução de automóveis de forma autônoma em rodovias, ferramentas que jogam xadrez de forma semelhante a campeões, entre outras (FACELI *et al.*, 2011). Os algoritmos de AM utilizados no trabalho são apresentados a seguir.

2.3.1.1 Árvore de Decisão

Segundo Russel e Norvig (2013), a indução de árvores de decisão é uma das formas mais simples, e ainda assim mais bem-sucedidas de aprendizagem de máquina. Uma árvore de decisão representa uma função que toma como entrada um vetor de valores de atributos e retorna uma "decisão", um valor de saída único. Os valores de entrada e saída podem ser discretos ou contínuos.

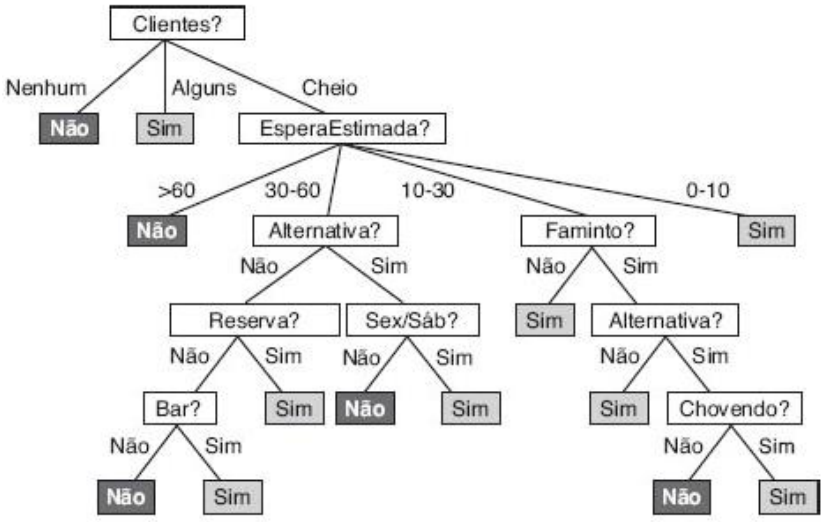
Kelleher *et al.* (2015) reforça que o processo de usar uma árvore de decisão para fazer uma previsão para uma instância de consulta começa testando o valor do recurso descritivo no nó raiz da árvore. O resultado deste teste determina para qual dos filhos do nó raiz o processo deve descer. Essas duas etapas de testar o valor de um recurso descritivo e descer um nível na árvore são então repetidas até que o processo chegue a um nó folha no qual uma previsão possa ser feita.

A representação de árvores de decisão é muito natural para os seres humanos; na realidade, muitos manuais do tipo "como fazer" (por exemplo, para consertos de automóveis) são inteiramente escritos como uma única árvore de decisão. Como exemplo, veja uma árvore de decisão para decidir a espera ou não de uma mesa em um restaurante. Aqui, o objetivo é aprender uma definição para o predicado de objetivo *VaiEsperar*. Primeiro listaremos os atributos que vamos considerar como parte da entrada e na Figura 4 é apresentada a árvore de decisão correspondente (RUSSEL e NORVIG, 2013):

1. *Alternativa:* Se há um restaurante alternativo apropriado por perto.
2. *Bar:* Se o restaurante tem uma área de bar confortável onde se possa esperar.
3. *Sex/Sáb:* Verdadeiro às sextas e sábados.
4. *Faminto:* Se estamos com fome.

5. *Clientes:* Quantas pessoas estão no restaurante (os valores são: Nenhum, Alguns e Cheio).
6. *Preço:* A faixa de preços do restaurante ($, $$, $$$).
7. *Chovendo:* Se está chovendo do lado de fora.
8. Reserva: Se fizemos uma reserva.
9. *Tipo:* O tipo de restaurante (francês, italiano, tailandês ou só de hambúrguer).
10. *EsperaEstimada:* A espera estimada pelo gerente (0-10 minutos, 10-30, 30-60, >60).

Figura 4 - Árvore de Decisão se espera ou não por uma mesa



Fonte: Russel e Norvig (2013)

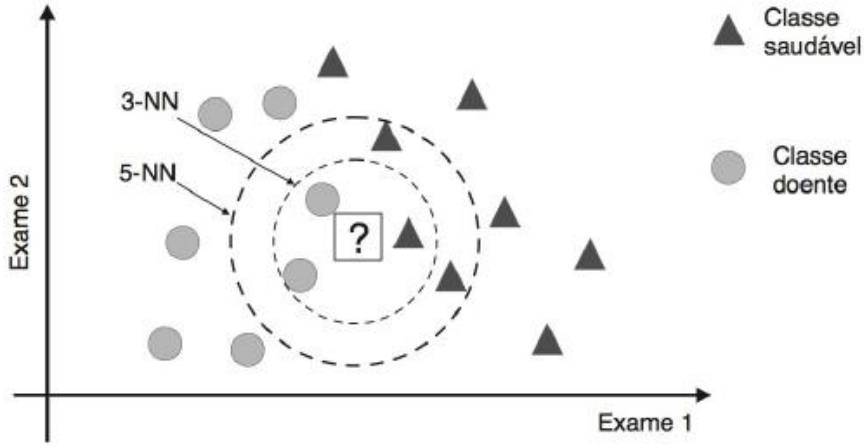
2.3.1.2 k-NN

O k-NN (*K-Nearest Neighbors*), conhecido como algoritmo dos vizinhos mais próximos, se encaixa nos métodos de similaridade, classificando um novo objeto com base nos exemplos de conjuntos de treinamento próximos a ele. É considerado um algoritmo preguiçoso porque não aprende um modelo compacto para os dados, apenas memoriza os objetos de treinamento. Uma vantagem desse algoritmo é que pode ser utilizado para problemas de classificação ou problemas de regressão de maneira direta, sem necessidade de grandes alterações (Faceli *et al*. 2011).

Faceli *et al.* (2011) reforça que no k-NN, para cada ponto de teste, são obtidos k vizinhos. Cada vizinho vota em uma classe, tal que as previsões dos diferentes vizinhos são agregadas de forma a classificar o ponto de teste. Considere o problema da Figura 5. Para k=3, o objeto de teste seria classificado como

pertencente à classe “doente”, enquanto para k=5 o objeto de teste seria classificado como pertencente à classe “saudável”.

Figura 5 – Impacto do valor de k do algoritmo k-NN



Fonte: Faceli *et al*. (2011)

A escolha do valor de k mais apropriado é definido pelo usuário, sendo frequentemente um valor pequeno e ímpar, por exemplo, k=3,5. Em problemas de classificação, não é usual k = 2 ou valores pares, para evitar empates.

2.3.1.3 Naive Bayes

O Classificador Naive Bayes é definido com base no teorema de Bayes, uma equação que é a base de todos os sistemas modernos de IA para inferência probabilística (RUSSEL e NORVIG, 2013). O Teorema é exibido na Figura 6.

Figura 6 – Teorema de Bayes

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Fonte: Faceli et al. (2011)

Ela nos permite calcular o único termo P (A | B) em torno de três termos: P (B | A), P (B) e P (A). P (B | A) busca responder qual é a probabilidade de B acontecer dado que o evento a já aconteceu; P (A | B) é a probabilidade do evento A ocorrer, considerando que o evento B já ocorreu; P (B) é a probabilidade do evento B, assim como P (A) é a probabilidade do evento A (RUSSELL E NORVIG, 2013).

A regra de Bayes é útil na prática porque existem muitos casos em que fazemos boas estimativas de probabilidade para esses três números e precisamos

calcular o quarto número. Muitas vezes, percebemos o efeito como evidência de alguma causa desconhecida e gostaríamos de determinar essa causa. Nesse caso, a regra de Bayes torna-se:

Figura 7 – Teorema de Bayes

$$P(causa \mid efeito) = \frac{P(efeito \mid causa)\, P(causa)}{P(efeito)}$$

Fonte: Facelli et al. (2011)

A probabilidade condicional P (efeito | causa) quantifica a relação na direção causal, enquanto P (causa | efeito) descreve a direção do diagnóstico. Em uma tarefa como o diagnóstico médico com frequência temos probabilidades condicionais sobre relacionamentos causais (isto é, o médico conhece P (sintomas | doenças)) e quer derivar um diagnóstico P (doenças | sintomas) (RUSSELL e NORVIG, 2013).

Por exemplo, um médico sabe que a meningite faz o paciente ter uma rigidez no pescoço, digamos, durante 70% do tempo. O médico também conhece alguns fatos incondicionais: a probabilidade a priori de um paciente ter meningite é 1/50.000, e a probabilidade a priori de qualquer paciente ter rigidez no pescoço é 1%. Sendo s a proposição de que o paciente tem rigidez no pescoço e m a proposição de que o paciente tem meningite, temos:

Figura 8 – Teorema de Bayes

$$\begin{aligned} P(s \mid m) &= 0{,}7 \\ P(m) &= 1/50000 \\ P(s) &= 0{,}01 \\ P(m \mid s) &= \frac{P(s \mid m)P(m)}{P(s)} = \frac{0{,}7 \times 1/50000}{0{,}01} = 0{,}0014\ . \end{aligned}$$

Fonte: Russell e Norvig (2013)

Ou seja, esperamos que apenas um em 5.000 pacientes com rigidez no pescoço tenha meningite. Note que, embora a rigidez no pescoço seja uma indicação bastante forte de meningite (com probabilidade 0,7), a probabilidade de o paciente estar acometido de meningite permanece pequena. Isso ocorre porque a probabilidade a priori sobre rigidez no pescoço é muito mais alta que a probabilidade a priori sobre meningite.

2.3.1.4 XGBoost

Chen e Guestrin (2016) criaram o XGBoost, um algoritmo de aprendizado de máquina baseado em árvore de decisão que usa um *gradient boosting framework*. Morde (2019) defende ser uma combinação perfeita de técnicas de otimização de software e hardware para produzir resultados superiores usando menos recursos de computação no menor espaço de tempo.

XGBoost e *Gradient Boosting Machines* (GBMs) são métodos de árvore de conjunto que aplicam o princípio de impulsionar aprendizes fracos usando a arquitetura de gradiente descendente. No entanto, o XGBoost melhora a estrutura GBM de base por meio da otimização de sistemas e aprimoramentos algorítmicos, conforme a Figura 9 (MORDE, 2019).

Figura 9 – Como o XGBoost otimiza o GBM padrão

Fonte: Adaptado de Morde (2019)

Padrões de acesso ao cache, compactação e fragmentação de dados são elementos essenciais para a construção de um sistema ponta a ponta escalonável para aumento de árvore. Ao combinar esses insights, o XGBoost é capaz de resolver problemas de escala do mundo real usando uma quantidade mínima de recursos (CHEN e GUESTRIN, 2016). Morde (2019), complementa que o XGBoost tem se destacado pois:

- Pode ser usado para resolver problemas de regressão, classificação, classificação e previsão definida pelo usuário;
- É portável e funciona perfeitamente no Windows, Linux e OS X;

- Suporta todas as principais linguagens de programação, incluindo C++, Python, R, Java, Scala e Julia;
- Devido a sua integração na nuvem, oferecendo suporte a clusters AWS, Azure e Yarn e funcionando bem com Flink, Spark e outros ecossistemas.

2.4 Aprendizado de máquina nas estimativas de esforço de software

Diante da infinidade de áreas que AM pode ser aplicada, pesquisadores passaram a inseri-lo nas estimativas de esforço de software. Na Revisão Sistemática realizada por Wen *et al.* (2012) abordando estudos entre 1990 e 2010, oito tipos de técnicas foram utilizados: Raciocínio Baseado em Casos, Redes Neurais Artificiais, Árvores de Decisão, Redes Bayesianas, Regressão vetorial de suporte, Algoritmos Genéticos, Programação Genética e Regras de Associação. Afirmam também, que devido aos modelos de AM serem orientados a dados tanto na construção quanto na validação, no momento de validá-los é essencial considerar os processos de obtenção destes dados. Modelos de AM vem sendo utilizados nas estimativas de esforço, porém, sua eficiência depende da qualidade e quantidade dos dados armazenados nas organizações, uma cultura ainda recente nas empresas de TI.

Além disso, a pesquisa revelou poucos estudos abordando a aplicação de modelos de AM na indústria, recomendando a condução de estudos visando descobrir barreiras para a adoção neste contexto. Aos profissionais o trabalho fornece informações sobre a aplicação dos modelos na indústria. São elas: (1) apresenta a lógica dos modelos e fatores de cooperação com pesquisadores experientes na aplicação de modelos de AM; (2) recomenda a escolha de modelos cujos pontos fortes combinam com os contextos de estimativas e metas; (3) propõe que dê prioridade aos conjuntos de dados dentro da empresa construindo modelos, se houverem dados suficientes; (4) aconselha a utilização dos modelos em paralelo com os modelos convencionais em fase inicial; e (5) reforça para o compartilhamento de conjuntos de dados de projeto proprietários com a comunidade de pesquisa (WEN et al., 2012).

Em Revisão Sistemática, Sharma e Singh (2017) concluem que uma quantidade significativa de pesquisas foi realizada em estimativa de esforço de software usando abordagens de AM. As principais abordagens utilizadas são Redes

Neurais Artificiais, Lógica Fuzzy, Algoritmos Genéticos e Árvores de Regressão. A análise revelou ainda a falta de conjuntos de dados reais que estejam de acordo com os métodos atuais de desenvolvimento de software e, também a necessidade de outras métricas confiáveis que possam ser usadas. Além disso, as tendências de pesquisa indicam a necessidade de explorar e criar novas técnicas de estimativas.

Também em Revisão Sistemática, Dantas *et al.* (2018) observou um uso crescente de técnicas inteligentes para estimativa de esforço no desenvolvimento ágil de software. Os trabalhos usam dados históricos e conhecimento especializado para apoiar a tomada de decisão.

A utilização da IA por meio de seus modelos de AM é realidade nas estimativas de esforço de software podendo ser útil. Variações são testadas e técnicas combinadas visando utilizar de informações históricas da empresa em apoio a situações atuais e isso apresenta-se como uma alternativa ascendente.

2.5 Base de conhecimento de software

Base de conhecimento de software foi definida por Meyer (1985) como todas as informações úteis de um projeto. Ela é utilizada por gestores e programadores para manter informações importantes dos componentes de software, incluindo documentos dos projetos, tarefas, orçamentos, entre outros. Deve-se certificar que todos os que necessitam de informações da base tenham acesso ao conhecimento (RUS *et al.*, 2002). A estrutura e a forma como os dados estão organizados, muitas vezes é referida como base de experiência, de modo que, para gerir a base, geralmente utiliza-se ferramentas que apoiem na captura, armazenagem, integração e recuperação das informações dos projetos (BASILI *et al.*, 2001).

Segundo Provost e Fawcett (2013) com grandes quantidades de dados agora disponíveis, as empresas da maioria dos setores estão focadas na exploração de dados para vantagem competitiva. O volume e a variedade de dados ultrapassaram em muito a capacidade de análise manual, de modo que se criou a ciência de dados, um conjunto de princípios fundamentais que apoiam e orientam a extração de informações e conhecimento dos dados.

Importante destacar que esta base de dados (que pode ser convertida em conhecimento) só tem valor se tratada como uma entidade viva, sendo nutrida,

cuidada, cresça e renove-se. Quando não é mantida regularmente a confiança de suas informações diminui, fugindo de seu objetivo principal que é armazenar informações importantes de projetos e constantemente contribuir na melhoria da produção de software (BASILI *et al.*, 2001).

É fundamental a construção de bases de conhecimento nas instituições como um todo, em especial no cenário de software, podendo trazer grandes benefícios e melhorias institucionais. A chegada da ciência dados com seu objetivo de auxiliar na tomada de decisões ao invés do uso da pura intuição permite uma assertividade maior nos mais diversos contextos (PROVOST; FAWCETT, 2013).

3 TRABALHOS RELACIONADOS

O processo de combinação ou construção de modelos híbridos é comum nas diversas áreas de conhecimento. Por meio da combinação procura-se mitigar problemas apresentados em abordagens individuais, propondo alternativas que possam contribuir de maneira eficiente em contextos específicos. Sendo assim, o presente capítulo aborda circunstâncias em que técnicas foram combinadas, servindo de base e fortalecendo o objeto de estudo.

Na *survey* de Rastogi et al. (2014) foi abordado que estimativa do esforço de software continua sendo um problema complexo. Nas últimas décadas, técnicas foram implementadas e bons resultados obtidos, com taxas de erro aceitáveis. Todavia, concluem que, nenhuma técnica é excepcional quando implantada sozinha e que duas ou mais podem ser integradas criando um modelo híbrido permitindo eliminar a maioria das falhas. Complementam, “Toda técnica tem seus próprios méritos e armadilhas. Porém, não existe uma técnica única capaz de eliminar todas as deficiências e ser aceita globalmente; portanto, a hibridação de várias abordagens pode ser vista como uma alternativa para produzir estimativas realistas”.

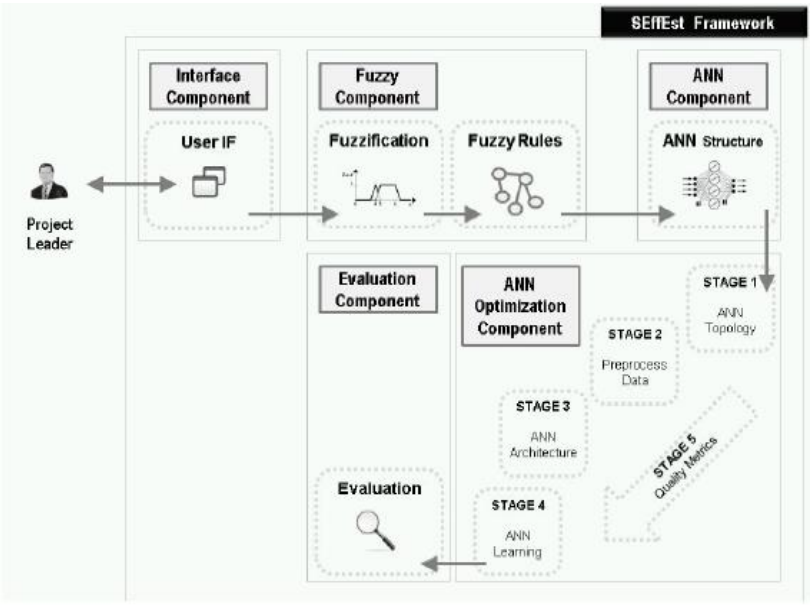
Segundo Elish (2013) nos últimos anos a estimativa do esforço usando modelos de inteligência computacional tem recebido aumento de atenção visando melhorar a precisão de estimativa. No entanto, nenhum dos modelos existentes provou ser adequado em todas as circunstâncias. O desempenho desses varia de um conjunto de dados para outro, havendo a necessidade de construir modelos que sejam confiáveis. Neste contexto, surgiu como alternativa a combinação de técnicas, também chamada de utilização das técnicas em conjunto. Por meio de uma pesquisa empírica o autor reforça que modelos utilizados em conjunto oferecem melhor desempenho do que modelos individuais.

Hosni *et al.* (2016) afirmaram que estimativas de esforço em conjunto vêm sendo investigadas e consistem em gerar o esforço combinando mais de uma técnica de estimativa unidas por uma regra de combinação. Corroborando, Kocaguneli *et al.* (2011) expuseram que os métodos usados em conjunto superam a maioria dos métodos individuais, tem desempenho mais confiável e evitam erros graves nas estimativas.

Bardsiri *et al.* (2012) relataram que um método de estimativa pode ser adequado apenas a um tipo especial de projeto de software e a dinâmica do desenvolvimento de software é complexa. Os autores propõem um método híbrido que consiste numa combinação de Raciocínio Baseado em Casos e Rede Neural Artificial. Inicialmente ocorre o processo de clusterização, em que ocorre a separação dos projetos em clusters de acordo com semelhanças existentes. Na sequência as técnicas são usadas em conjunto atuando de acordo com a quantidade de dados em cada cluster. Cluster de dados de baixa população usam Raciocínio Baseado em Casos e, por sua vez, Rede Neural Artificial é usada quando se tem aglomerados populacionais elevados, visto que terão melhor desempenho.

González-Carrasco et al. (2012) construíram um framework chamado SEffEst que é baseado em Lógica Fuzzy e Redes Neurais Artificiais. A Lógica Fuzzy e a Rede Neural são componentes do framework atuando em sequência conforme exibido na Figura 10. A Lógica Fuzzy é aplicada primeiro, pois é capaz de processar dados incompletos e fornece informações aproximadas para problemas que outros métodos acham difíceis de resolver. Na sequência a rede neural é inserida visando melhorar o desempenho da operação anterior em tempo e precisão. Realizando atividades de validação por meio do fator de correlação médio, obteve-se melhora de 6,15% em comparação a um trabalho anterior dos autores que não utilizada a Lógica Fuzzy no processo.

Figura 10 – SeffEst framework

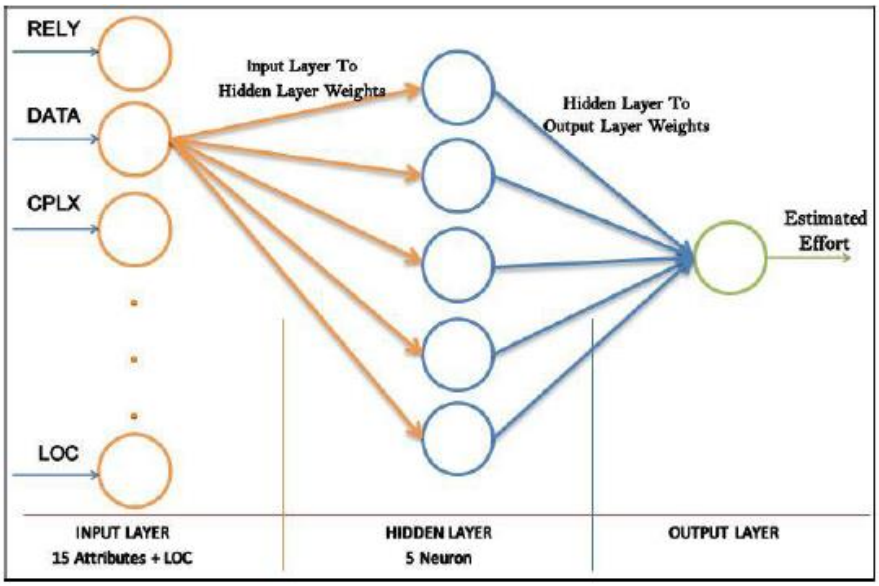


Fonte: González-Carrasco et al. (2012)

Na abordagem proposta por Pospieszny et al. (2015) ocorre a combinação de medição de tamanho funcional do software via Pontos de Função e três técnicas de AM: Redes Neurais Artificiais, Árvores de Decisão e Modelos Lineares Generalizados. A abordagem consiste em captar dados via Pontos de Função e após aplicar cada algoritmo separadamente, seus resultados são unidos posteriormente através de média das três técnicas. Os resultados apontam uma boa capacidade dos modelos individuais com base em dimensionamento de software avaliado com pontos de função. No entanto, a abordagem proposta combinando previsões gera ainda mais previsões precisas e acima de tudo superam a possibilidade de sobre ajuste e entregar previsões falsas por algoritmos individuais.

Segundo Saraç e Duru (2013) as estimativas são a base do planejamento, programação, orçamento e outras necessidades de contrato. Os autores abordam a combinação de COCOMO com Redes Neurais Artificiais, utilizando do seguinte processo: o conjunto de dados baseado na técnica COCOMO é usada como entrada da rede neural apresentada na Figura 11 e por fim, a saída da rede é usada como entrada para o K-Means usado para calcular os limites superior e inferior, também chamados de limites de esforço. O modelo proposto é significativamente melhor que o COCOMO ou Rede Neural Artificial quando utilizadas individualmente.

Figura 11 – Rede neural do modelo

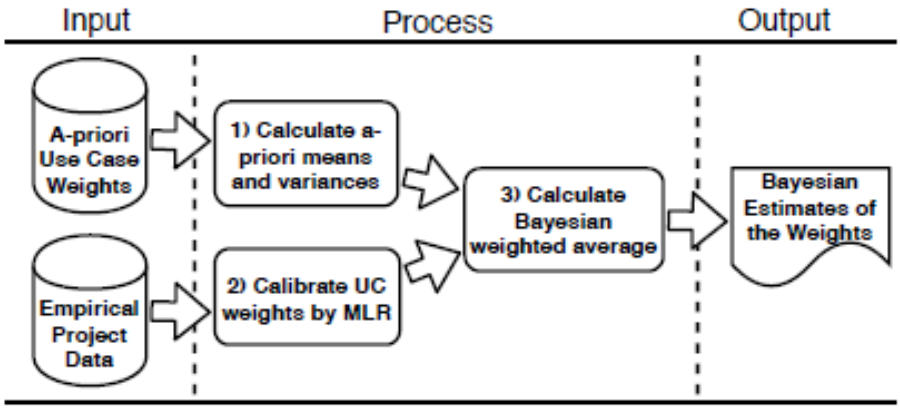


Fonte: Saraç e Duru (2013)

Os Pontos de Caso de Uso têm sido amplamente utilizados para estimar o tamanho do software em projetos orientados a objetos. No entanto, muitos trabalhos criticam sua metodologia por não serem verificados e validados com dados, levando

a imprecisão. Qi *et al.* (2018) exploraram o uso da Análise Bayesiana para calibrar os pesos de complexidade do caso de uso para melhorar precisão do tamanho do software e da estimativa do esforço. A Análise Bayesiana é aplicada integrando informações prévias com valores de parâmetros sugeridos por regressão linear múltipla. Os resultados mostram que estimativas bayesianas dos pesos de complexidade de casos de uso fornecem melhor precisão de estimativa em comparação aos pesos propostos pelo Pontos de Caso de Uso original. O esboço do modelo proposto é apresentado na Figura 12.

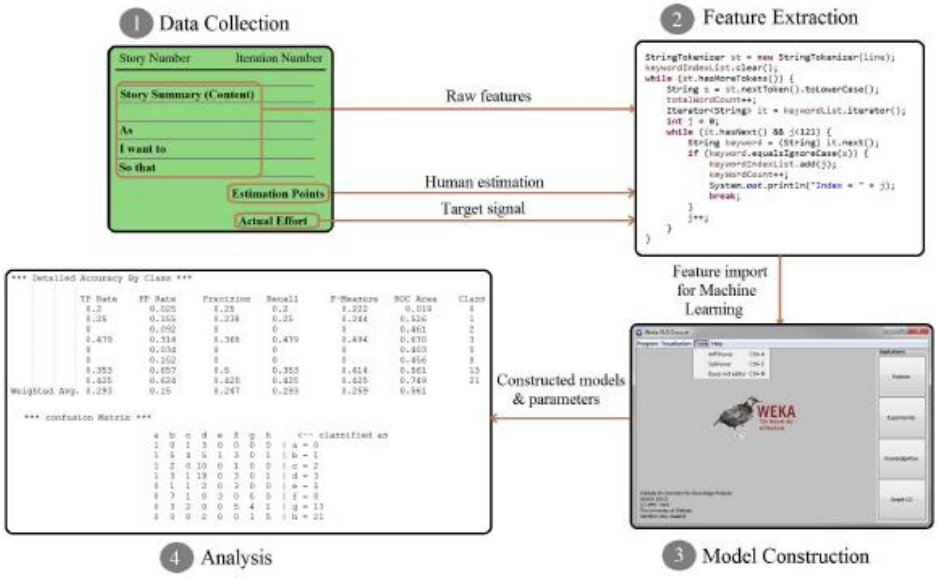
Figura 12 – Modelo proposto



Fonte: Qi et al. (2018)

No trabalho desenvolvido por Moharreri *et al.* (2016) criou-se um método de estimativa automatizada para cartões de história de usuário. Este método aprimora o *Planning Poker* manual, método de estimativa mais usado em ambientes ágeis. A estimativa automática utiliza recursos extraídos de fichas usadas num ambiente de desenvolvimento. O fluxo do método é apresentado na Figura 13 e os passos são os seguintes: (1) Coleta de Dados: usando fichas, estimativas humanas e dados de esforço reais; (2) Extração de recursos: extraindo tokens usando análise de texto; (3) Construção de modelo: usando recursos extraídos e estimativas humanas, juntamente com o esforço real para treinar classificadores; e (4) Análise: medir o desempenho para selecionar os melhores classificadores. A medição de desempenho ocorreu de modo que algumas abordagens combinavam as técnicas para estimar esforço e outras que utilizavam apenas as técnicas de classificação, sem o *Planning Poker* manual.

Figura 13 – Etapas da proposta



Fonte: Moharreri et al. (2016)

Nos trabalhos apresentados percebe-se que os tipos de combinações são diversos, envolvem características particulares e surgem como alternativa ao processo de estimativas. As experimentações envolvendo AM são diversas e apresentam-se na grande maioria dos estudos, reforçando a tendência de utilização no campo da engenharia de software e mais especificamente no processo de estimativas de esforço, objeto deste estudo. O Quadro 1 contempla as características chaves dos trabalhos abordados anteriormente.

Quadro 1 – Principais características dos trabalhos relacionados

Característica versus Referência	
Tipos de Técnicas Combinadas	**Referência**
Aprendizado Máquina + Aprendizado Máquina	Bardsiri et al. (2012), Gonzalez-Carrasco et al. (2012)
Aprendizado de Máquina + Modelo Algorítmico	Pospieszny et al. (2015), Saraç e Duru (2013), Qi et al. (2018)
Aprendizado de Máquina + Opinião Especializada	Moharreri et al. (2016).
Característica da Combinação	**Referência**
Híbrido	Bardsiri et al. (2012),
Sequencial	Gonzalez-Carrasco et al. (2012) Qi et al. (2018)
Dados de uma técnica servindo de entrada para a outra	Pospieszny et al. (2015), Saraç e Duru (2013), Moharreri et al. (2016)

Fonte: Autoria Própria (2020)

Nossa proposta corrobora com os trabalhos abordados pois defendemos que técnicas individuais tem situações que podem ser melhoradas por meio da combinação a outras técnicas. Além disso, assemelha-se a proposta de Moharreri *et al.* (2016) no ato de utilizar o *Planning Poker* (técnica de opinião especializa) e AM, porém diferenciam-se no fato de que Moharreri *et al.* (2016) usou processo de tokenização, o apoio da ferramenta Weka para os algoritmos de classificação. Por sua vez, em nossa abordagem utilizamos os dados separados por colunas (variáveis descritivas e variável alvo) e os algoritmos são usados através da biblioteca *Scikit-Learn* sendo executados na ferramenta Colab Notebook.

3.1 Análise dos principais problemas em aberto

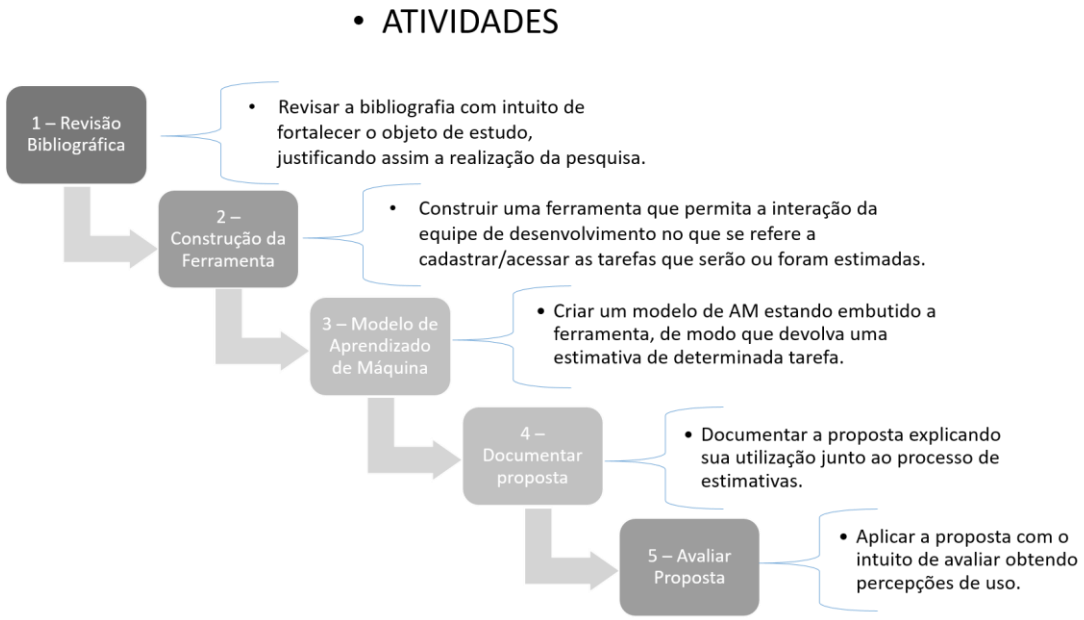
A exploração do campo de estudo permitiu coletar percepções de possíveis situações a serem exploradas, de modo que, cremos ser possível contribuir com o estado da arte e comunidade científica da área. Em âmbito mais amplo percebemos que o processo de estimativas de esforço de software continua oportunizando a exploração devido às incertezas associadas ao processo. Mais especificamente analisando o contexto do desenvolvimento ágil, desde 2014 a comunidade científica vem sendo muito ativa na busca de soluções mais precisas.

No que se refere a combinação de técnicas visando minimizar as falhas de estimativas de técnicas individuais e utilização de algoritmos de AM, encontramos trabalhos que abordam o tema e, também espaço para novos estudos na combinação do AM com o envolvimento do fator humano no processo. Neste contexto propomos a *ML Planning Poker*, com propósito de avaliar se a combinação do *Planning Poker* com AM interfere no processo de estimativas de esforço de software.

4 MATERIAIS E MÉTODOS

De encontro ao objetivo do trabalho, esta Seção descreve a estrutura e o método de trabalho utilizados. A Figura 14 apresenta as 5 etapas e suas atividades. São elas: 1) revisão bibliográfica; 2) construção da ferramenta; 3) construção do modelo de aprendizado de máquina; 4) documentar proposta; e, 5) avaliar proposta.

Figura 14 - Etapas da proposta



Fonte: Autoria Própria (2020)

Inicialmente realizamos pesquisa bibliográfica em busca de informações que argumentassem sobre o problema de pesquisa, referencial teórico e trabalhos relacionados. Nesta etapa também, construímos um mapeamento sistemático e uma *survey* fortalecendo as bases do estudo, permitindo a percepção de problemas em aberto com possibilidade de exploração.

Conhecendo melhor a área, propomos a *ML Planning Poker*, combinação do *Planning Poker* com AM, visando descobrir como a abordagem comporta-se no processo de estimativas de esforço de software. Nesta direção o próximo passo consistiu da criação de uma ferramenta servindo de apoio e meio interativo com as equipes de desenvolvimento no processo de estimativas, tal que embutido a ela criamos um modelo de AM. Na sequência documentamos a proposta objetivando esclarecer de que forma a abordagem é inserida ao contexto. Por fim, realizamos a avaliação da abordagem com intuito de analisar e testar seu desempenho. Cada etapa é descrita em detalhes a seguir.

4.1 Revisão bibliográfica

Realizamos uma pesquisa bibliográfica utilizando palavras-chaves relacionadas ao tema, como: estimativa de esforço de software, *Planning Poker*, aprendizado de máquina e combinação de técnicas de esforço de software. De acordo com Lima e Mioto (2017), a pesquisa bibliográfica é executada para fundamentar o objeto de estudo, contribuindo com elementos que amparam a análise futura dos dados coletados. Os autores ressaltam também, que é um procedimento metodológico importante na produção do conhecimento científico capaz de gerar a postulação de hipóteses ou interpretações, servindo de ponto de partida para outras pesquisas.

Na pesquisa bibliográfica buscamos artigos que abordassem a utilização do *Planning Poker*, a aplicabilidade de técnicas de IA ou AM nas estimativas de esforço de software, entre outros. O intuito foi identificar especificidades ou relações entre os temas e hipóteses fossem obtidas fortalecendo a relevância do estudo.

Além da revisão bibliográfica, produzimos um mapeamento sistemático e uma *survey* permitindo explorar mais profundamente o campo de estudo. O mapeamento sistemático ocorreu em torno da combinação de técnicas para estimar esforço de software permitindo a obtenção de percepções sobre o uso da combinação de técnicas. Por meio dele encontramos os trabalhos relacionados apresentados no Capítulo 3. A *survey* permitiu instigar os profissionais de TI que realizam estimativas de esforço de software nas cidades de Chapecó e Curitiba, e assim descobrimos as principais técnicas, entre outros aspectos referentes à utilização e percepção da combinação de técnicas.

A relevância da exploração do tema via literatura e também em contato com os profissionais permitiu compreender melhor a inserção da pesquisa na prática. Os principais resultados do mapeamento e *survey* serão exibidos na Seção 5.1.

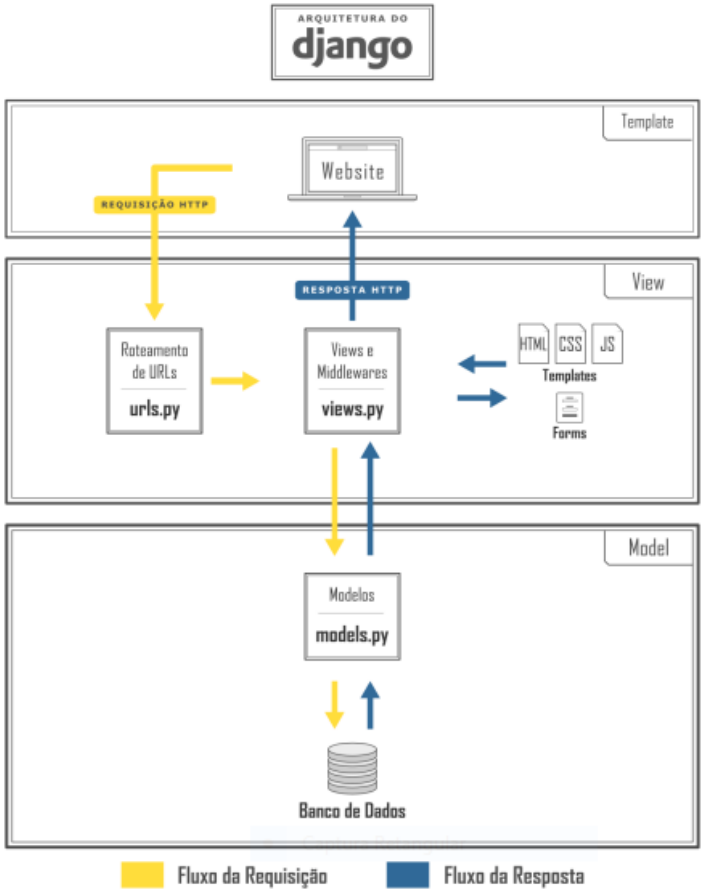
4.2 Construção da ferramenta

Construímos uma ferramenta disponível ao processo de estimativas, servindo de interface para o moderador do processo inserir os dados da tarefa que está sendo estimada. A partir dos dados inseridos, os mesmos são enviados

servindo de entradas ao modelo de AM que está embutido a aplicação, fornecendo uma estimativa com base em seu aprendizado.

Para a construção da ferramenta utilizamos o Django[1], um framework para o rápido desenvolvimento de sites seguros e de fácil manutenção, escrito em Python que utiliza o padrão *model-template-view* (MTV). É gratuito, de código aberto, tem comunidade ativa, documentação abrangente e muitas opções de suporte. A Figura 15 apresenta a arquitetura do framework.

Figura 15 – Arquitetura do Django



Fonte: (RAMOS, 2018)

Referindo-se ao banco de dados, optamos pelo PostgreSQL[2], um sistema de banco de dados objeto-relacional de código aberto e desenvolvimento ativo.

Para o *front-end* utilizamos o *Bootstrap*[3] que é um framework web com código-fonte aberto para desenvolvimento de componentes de interface e *front-end*

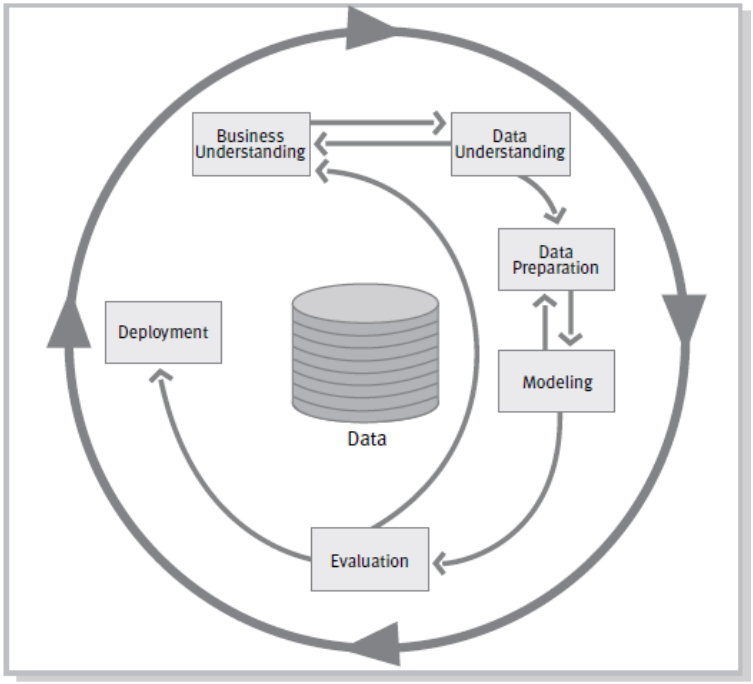
[1] https://www.djangoproject.com/
[2] https://www.postgresql.org/
[3] https://getbootstrap.com/

para sites e aplicações web usando HTML, CSS e JavaScript. A Seção 5.2 aborda aspectos referentes a construção da ferramenta.

4.3 Modelo de aprendizado de máquina

Para a construção do modelo de AM usamos o modelo de processo *Cross Industry Standard Process for Data Mining* (CRISP-DM) um modelo difundido servindo de ciclo de vida em projetos de *data mining*. Proposto por Chapman *et al.* (2000), consiste de 6 fases como apresentado na Figura 16. A sequência das fases não é rígida e mover para frente/para trás é sempre requerido. As setas indicam as mais frequentes dependências entre fases, sendo importante destacar também o círculo mais externo que simboliza a natureza cíclica de mineração de dados em si.

Figura 16 – Fases do Modelo de Referência CRISP-DM



Fonte: (CHAPMAN *et al.*, 2000)

A seguir é descrito brevemente cada fase (CHAPMAN *et al*., 2000):

- **Compreensão do negócio (*Business understanding*):** Essa fase refere-se ao entendimento dos objetivos e requisitos do projeto de uma perspectiva comercial, depois ocorre a conversão desse conhecimento numa definição de problema e num plano preliminar desenvolvido para atingir os objetivos.
- **Compreensão dos dados (*Data understanding*):** A fase de entendimento dos dados começa com a coleta inicial de dados e prossegue com as

atividades que permitem familiarizar-se com os dados, identificar problemas de qualidade ou imperfeições, descobrir as primeiras informações ou detectar subconjuntos interessantes para formar hipóteses sobre informações ocultas.

- **Preparação dos dados (*Data preparation*):** A fase de preparação abrange as atividades necessárias para construir o conjunto de dados final (que serão alimentados nas ferramentas de modelagem) a partir dos dados brutos iniciais. É provável que as tarefas de preparação ocorram várias vezes e não numa ordem prescrita. As tarefas incluem seleção de tabela, registro e atributo, além de transformação e limpeza de dados.
- **Modelagem (*Modeling*):** Nesta fase, várias técnicas de modelagem são selecionadas e aplicadas, e seus parâmetros calibrados para valores ideais. Normalmente, existem várias técnicas para o mesmo tipo de problema de mineração de dados. Algumas têm requisitos específicos na forma de dados, assim, muitas vezes é necessário voltar à fase de preparação.
- **Avaliação (*Evaluation*):** Nesta fase você construiu um modelo (ou modelos), mas antes de prosseguir para a implantação, é importante avaliá-lo completamente e revisar as etapas executadas para criá-lo, garantindo que o modelo atinja adequadamente os objetivos. No final desta fase, uma decisão sobre o uso dos resultados da mineração de dados deve ser alcançada.
- **Implantação (*Deployment*):** A criação do modelo geralmente não é o fim do projeto. Mesmo que o objetivo do modelo seja aumentar o conhecimento dos dados, o conhecimento adquirido precisará ser organizado e apresentado de forma que o cliente possa usá-lo. É importante que este entenda antecipadamente quais ações precisam ser executadas para realmente fazer uso dos modelos criados.

Para a construção do modelo de AM utilizamos a ferramenta Colab Notebook[4], na linguagem de programação Python com o uso das bibliotecas Pandas[5] (análise e manipulação de dados), *Numpy*[6] (transformação de dados), *Matplotlib*[7] (visualização de dados), *Scikit-Learn*[8] (modelos e testes estatísticos) e *XGBoost*[9] (framework de *gradient boosting*).

[4] https://colab.research.google.com/notebooks/welcome.ipynb?hl=pt-BR
[5] https://pandas.pydata.org/
[6] https://numpy.org/
[7] https://matplotlib.org/

Os dados para a construção do modelo foram buscados em repositórios públicos, sendo selecionado um *dataset* (conjunto de dados) contendo a descrição de tarefas de software (suas variáveis descritivas), e o tempo de implementação atuando como variável alvo. Após a preparação dos dados, foram aplicados algoritmos de AM diversificados como por exemplo: Árvores de Decisão, Naive Bayes e k-NN. Aplicados os algoritmos, analisamos seu desempenho sobre os dados, sendo utilizada como métrica de performance o relatório gerado pelo *classification_report* da biblioteca *sklearn.metrics*. A Seção 5.3 apresenta as etapas de criação do modelo de AM com os resultados.

4.4 Documentar proposta

A ferramenta e o modelo de AM embutido a ela servem de apoio ao processo de estimativa de esforço de software com foco em equipes ágeis e que utilizam o *Planning Poker*. O processo de documentação é importante para esclarecer sobre a utilização da *ML Planning Poker* sem que ocorram dúvidas ou ambiguidades e assim seja possível analisá-la na íntegra e sem vieses. A documentação ocorreu num contexto teórico no que diz respeito aos procedimentos e definições a serem realizadas obedecendo o rigor científico e também voltada ao caráter prático referindo-se à aplicação em cenários reais.

4.5 Avaliar proposta

O processo de avaliação da *ML Planning Poker* consiste em aplicá-lo a um contexto real para avaliar seu desempenho. Avaliamos através de abordagem híbrida envolvendo pesquisa qualitativa e quantitativa. Acreditamos que desta maneira valorizam-se as percepções humanas, sem esquecer da análise de desempenho proporcionada pelas técnicas estatísticas.

---

[8] https://scikit-learn.org/stable/
[9] https://xgboost.readthedocs.io/en/stable/

Segundo Gerhardt e Silveira (2009), na pesquisa qualitativa preocupa-se com a realidade que não pode ser representada numericamente, centrando-se na compreensão e explicação da dinâmica das relações sociais. Suas características envolvem a objetivação do fenômeno e exige ações como descrever, compreender e explicar, buscando os dados mais fidedignos possíveis. Por sua vez a pesquisa quantitativa tem raízes no pensamento positivista lógico, tende a enfatizar o raciocínio dedutivo e os atributos mensuráveis da experiência humana.

Lakatos e Marconi (2007) reforçam que três traços bem definidos no conteúdo quantitativo são observados: objetividade, sistematização e quantificação dos conceitos. Concluem que no método quantitativo os pesquisadores valem-se de amostras amplas e de informações numéricas, enquanto no qualitativo as amostras são reduzidas, os dados são analisados em seu conteúdo psicossocial e os instrumentos de coleta não são estruturados.

Devido à dificuldade de implantação numa empresa perante as questões burocráticas e de acesso a informação, aplicamos a *ML Planning Poker* em ambiente acadêmico com alunos de graduação dos cursos de Sistemas de Informação e Engenharia da Computação da UTFPR, Campus Curitiba. Simulamos o cenário de equipes de desenvolvimento que estavam aprendendo sobre estimativas de esforço de software. Segundo Gerhardt e Silveira (2009), esta pesquisa pode ser definida de natureza aplicada, objetivando gerar conhecimentos para aplicação prática. Quanto aos procedimentos é classificada como experimental, desenvolvida em laboratório, pois o meio ambiente criado é artificial, neste caso a sala de aula. A abordagem utilizada foi antes-depois com um único grupo. Primeiro aplicamos o *Planning Poker* original e depois a *ML Planning Poker*. Por fim, coletamos as percepções sobre a abordagem através de questionário online.

Realizamos também a avaliação com um grupo de profissionais de TI que trabalham com estimativas e utilizam a prática *Planning Poker*. Para esta etapa, foram buscados profissionais voluntários, organizada uma reunião de apresentação da proposta e por fim coletadas as percepções dos participantes via questionário. O questionário respondido por estudantes e profissionais encontra-se no Apêndice A.

5 RESULTADOS E DISCUSSÕES

Seguindo a metodologia definida no capítulo anterior, neste capítulo são apresentados os resultados e discussões do trabalho.

5.1 Revisão bibliográfica

Consiste da etapa de exploração do tema argumentando sobre a relevância e proporcionando a base aos leitores sobre o referencial teórico envolvido ao estudo.

5.1.1 Mapeamento Sistemático

O mapeamento sistemático objetivou identificar trabalhos que abordassem a combinação de técnicas na estimativa de esforço de software, publicados entre 2008 e 2019. Por meio deste, analisamos canais e frequência das publicações, autores mais influentes, técnicas investigadas e combinadas, além das características das combinações.

A definição da *string* de busca ocorreu com base em palavras chaves referentes ao tema de pesquisa, tal que foram combinadas utilizando-se dos operadores lógicos AND e OR. A *string* de busca é apresentada a seguir: *"Software engineering" AND effort AND (estimates OR estimation OR prediction) AND (ensemble OR combining OR combination) AND ("machine learning" OR "artificial intelligence" OR "case based reasoning" OR "decision tree" OR "neural network" OR "expert opinion" OR "planning poker" OR "algorithmic models" OR "cocomo" OR "function points analysis")*.

A estratégia de pesquisa foi por meio de busca automática individual em cada uma das bases: *IEEE Xplore, ACM Digital Library, Scopus e Web Of Science*. Primeiramente foi executada uma pesquisa individual usando a *string* de pesquisa em cada uma das bases de dados. A partir daí o processo de seleção ocorreu em duas etapas: seleção inicial e seleção final. A Tabela 1 explicita os dados sobre o processo de busca e seleção.

Tabela 1 – Busca e Seleção dos Artigos

Bases de Busca	Trabalhos Encontrados	Análise de Título e Resumo	Seleção Inicial	Seleção Final
IEEE Xplore	34	34	8	3
ACM	30	30	5	1
Scopus	1701	1701	43	15
Web Of Science	59	59	4	0
Total	**1824**	**1824**	**60**	**19**

Fonte: Autoria Própria (2020)

Foram 19 estudos selecionados, permitindo perceber a relevância do tema, surgindo como alternativa aos problemas vivenciados pelas técnicas de estimativa individuais e apresentando-se como um tema explorado recentemente. Além disso, há grande diversidade nas técnicas investigadas e combinadas, com destaque para as técnicas de aprendizado de máquina que foram exploradas em todos os trabalhos selecionados conforme apresentado na Tabela 2.

Tabela 2 - Tipos de Técnicas Combinadas

Tipos de Técnicas Combinadas	Número de trabalhos
Aprendizado de Máquina e Modelos Algorítmicos	5
Aprendizado de Máquina e Opinião Especializada	1
Aprendizado de Máquina e Aprendizado de Máquina	13
Total	19

Fonte: Autoria Própria (2020)

A variedade de técnicas utilizadas e suas combinações cada qual com suas especificidades e maneira de mitigar os problemas, mostra que a combinação não se restringe a um conjunto limitado de técnicas. Percebemos a utilização das técnicas de forma independente, unindo apenas os resultados; a utilização dos dados levantados por uma técnica sendo utilizado de entrada em outra; por fim, técnicas utilizadas numa sequência ou até mesmo de forma híbrida em que aspectos interessantes de uma são utilizados embutidos a outra.

O mapeamento nos permitiu descobrir como a combinação de técnicas vem sendo explorada, proporcionando uma visão sobre as percepções de outros trabalhos em torno do tema. Os resultados auxiliaram na construção do Capítulo 3.

5.1.2 Survey

O intuito deste estudo foi identificar aspectos referentes à realização de estimativas de esforço de software e descobrir sobre a possibilidade de combinar mais de uma técnica para estimar, através do comparativo entre as cidades de Chapecó e Curitiba. Para atingir o objetivo realizamos um planejamento em que foram caracterizados o público alvo, o quadro da amostra, projetado e escrito o questionário e por fim definida a estratégia de recrutamento.

- **Caracterização do Público Alvo:** Profissionais da área de TI que realizam estimativas de esforço de software em seu cotidiano.
- **Quadro da Amostra:** Profissionais da rede de contatos dos pesquisadores que atuam em Chapecó ou Curitiba.
- **Escrita do Questionário:** Questões de caracterização do sujeito e investigativas.
- **Estratégia de Recrutamento:** Envio do link do Google Forms. O envio ocorreu para pessoas de Chapecó e Curitiba com as quais os pesquisadores possuem contato.

Definido o planejamento, a coleta das respostas ocorreu entre o período de 11/11/2019 a 20/11/2019, de modo que o número total de respostas foi 49. Do total, 23 respostas de Chapecó, 22 de Curitiba e 4 de outras cidades, pois não havia restrição de receber respostas de outras cidades. Para análise dos dados optamos em explorar de maneira geral todas as respostas, mas cientes da atenção especial para análises entre Chapecó *versus* Curitiba, objetivo geral do estudo.

Abordando a assertividade das estimativas realizadas (Questão 7) - Figura 17, a maioria delas foi definida como Boa (51%), seguido de Muito Boa (27%) e Regular (14%).

Figura 17 – Assertividade das Estimativas

Fonte: Autoria Própria (2020)

A maioria dos participantes (38 dos 49) nunca utilizou combinação de técnicas para estimar. Algumas combinações foram realizadas, como por exemplo: *Planning Poker* com Analogia e *Use Case Points* com Analogia apareceram em duas respostas cada. Outras combinações apareceram com uma resposta cada.

No que se refere a possibilidade de combinar técnicas a questão 10 foi: "Como você vê a possibilidade de combinar mais de uma técnica de estimativa?". A maioria dos participantes vê com bons olhos a possibilidade de combinação de técnicas, mas apareceram também respostas contrárias e alguns impedimentos foram levantados. Destacam-se algumas das respostas favoráveis no Quadro 2.

Quadro 2 – Respostas favoráveis a combinação de técnicas

Respostas
Com otimismo, uma vez que utilizamos em nosso processo de estimativa.
Válida. Contudo, deve ficar claro aos membros da equipe o que se almeja com as estimativas e como cada uma das técnicas funciona (caso contrário pode ocorrer manipulações, desencontros e situações indesejadas). Ainda, as técnicas de estimativa devem ser aderentes à maturidade das equipes e deve-se prover tempo suficiente para que os resultados da utilização de cada técnica possam ser analisados.
Excelente estratégia! Historicamente, equilibrar o uso híbrido de técnicas tem em geral dado bons resultados seja em TI ou em outras áreas. Inclusive pode ser uma linha de pesquisa interessante. Diga-se de passagem, as técnicas de IA para inferência à tomada de decisão e mesmo algo como fusão de dados, onde desvios podem ser corrigidos. Em resumo: a possibilidade de combinar mais de uma técnica pode reduzir distorções em pontos diferentes que cada uma das técnicas podem causar, uma auxiliando a outra.
Acho muito viável combinar mais de uma forma, estimando primeiramente em cada uma delas a fim de comparara diferença entre ambas, caso essa diferença de estimativa de tempo for grande provavelmente é um sinal que o requisito precisa ser melhor detalhado e feito uma nova estimativa.
Vejo como muito positiva. É possível unir Planning Poker com Ideal Day, mas também é possível introduzir futuramente alguma técnica de IA para análise do histórico de estimativas para auxiliar as técnicas executadas manualmente pelo time.
Possível, desde que não demande muito tempo.

Fonte: Autoria Própria (2020)

As constatações reforçaram que o processo ainda é falho, haja vista a quantidade de estimativas que são ruins ou regulares. Mesmo que a maioria dos profissionais ainda não tenham combinado técnicas para estimar esforço, esta possibilidade foi vista de maneira positiva por grande parte dos entrevistados e pode ser uma alternativa a ser explorada na busca por evolução no processo de estimativas frente as necessidades presenciadas.

5.2 Construção da ferramenta

Separamos a construção da ferramenta em duas etapas. A primeira referente às atividades de levantamento de requisitos e modelagem do sistema, enquanto na segunda etapa ocorreu a implementação.

5.2.1 Modelagem

Inicialmente definimos os requisitos e outras atividades referentes a modelagem de sistemas. Os Quadros 3 e 4 apresentam os Requisitos-Funcionais (RFs) e Requisitos Não-Funcionais (RNFs) da aplicação.

Quadro 3 – Requisitos funcionais do sistema

ID	Descrição
RF01	**Criar Cadastro:** Cada usuário necessita criar um cadastro, indicando um usuário e senha de acesso ao sistema.
RF02	**Efetuar Login:** Cada usuário necessita inserir seus dados previamente cadastrados para acessar o sistema.
RF03	**Vincular Integrante:** Cada integrante participa de uma equipe e deve ser vinculada a ela via administrador.
RF04	**Manter Tarefa:** Toda tarefa deve ser inserida e pode ser alterada, selecionada ou deletada.
RF05	**Manter Tarefas do Time:** Permite que conclua, altere ou selecione uma tarefa do time.
RF06	**Buscar Estimativa:** Buscar estimativa no modelo inteligente sobre a tarefa que está sendo estimada, bem como devolver uma lista com tarefas semelhantes, com base no verbo.
RF07	**Visualizar Sobre:** Tela que apresenta os envolvidos na pesquisa.
RF08	**Visualizar Manual do Usuário:** Tela que explica o funcionamento de cada funcionalidade.

Fonte: Autoria Própria (2020)

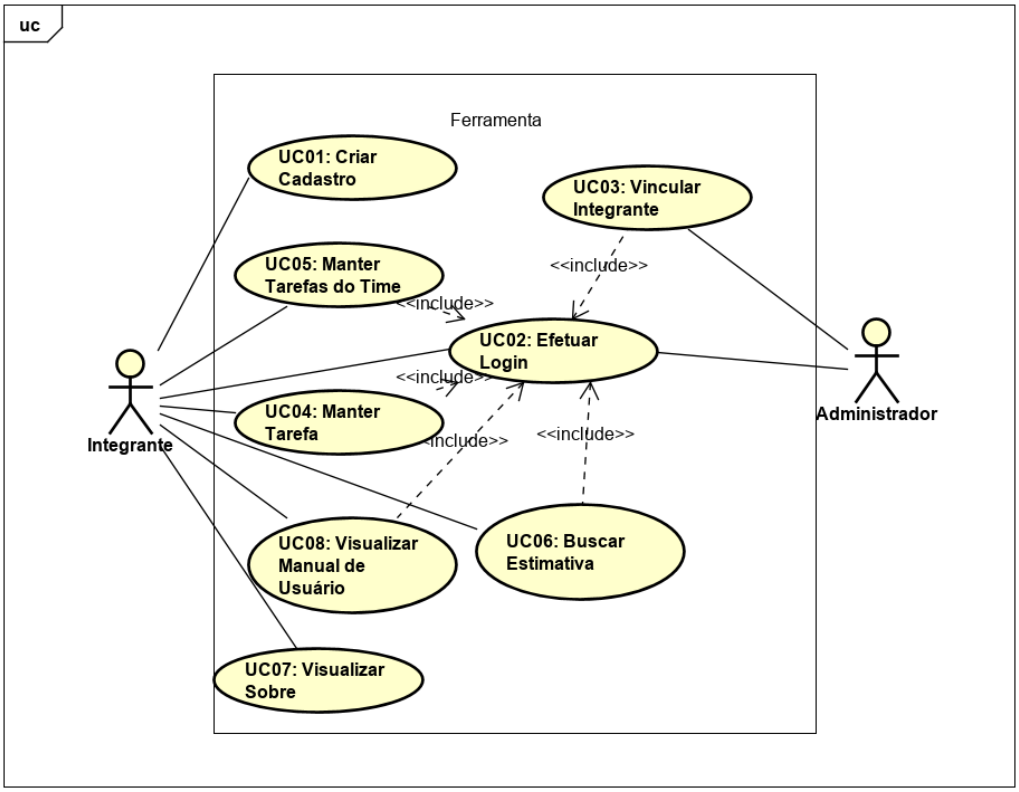
Quadro 4 – Requisitos Não-Funcionais do sistema

ID	Descrição
RNF01	**Banco de Dados:** O sistema utilizará banco de dados relacional.
RNF02	**Compatibilidade:** O sistema deve ser rodar no navegador Google Chrome.
RNF03	**Performance:** O tempo de resposta das requisições não pode ser superior a 5 segundos.
RNF04	**Usabilidade:** A interface deve ser atrativa usando cores e de fácil entendimento.

Fonte: Autoria Própria (2020)

A partir da identificação dos requisitos funcionais, definimos os casos de uso do sistema (*Use Cases* (UC)). Para facilitar a visualização e compreensão, construímos o diagrama de casos de uso (Figura 18) seguindo o padrão UML (*Unified Model Language*) utilizando a ferramenta Astah UML[10].

Figura 18 – Diagrama de Casos de Uso



Fonte: Autoria Própria (2020)

Com base nos RFs e UCs é possível relacioná-los permitindo rastrear o ciclo de vida dos requisitos de forma completa, a relação entre eles é chamada de matriz de rastreabilidade para frente. O Quadro 5 apresenta a matriz correspondente.

[10] https://astah.net/downloads/

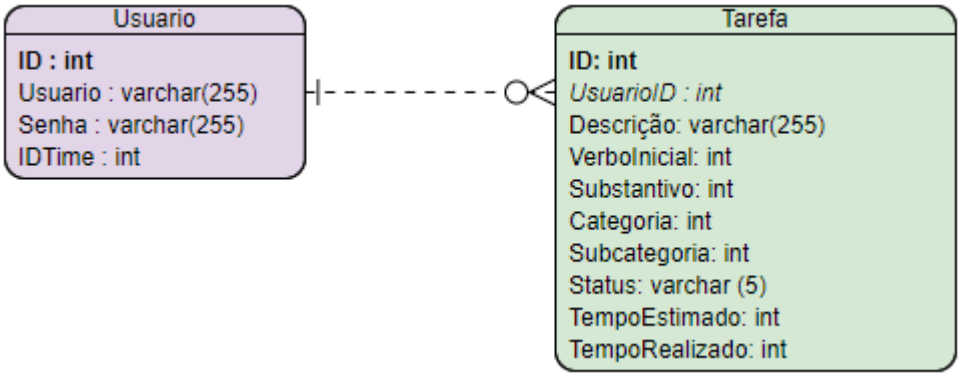
Quadro 5 – Matriz de rastreabilidade para frente

Req. / UseC.	RF01	RF02	RF03	RF04	RF05	RF06	RF07	RF08
UC01	X							
UC02		X						
UC03			X					
UC04				X				
UC05					X			
UC06						X		
UC07							X	
UC08								X

Fonte: Autoria Própria (2020)

A Figura 19 apresenta o diagrama Entidade-Relacionamento (ER) da ferramenta proposta, construída na ferramenta Visual Paradigm Online[11]. Por meio deste diagrama é possível visualizar as tabelas, indicar os campos, estabelecer as relações e suas cardinalidades.

Figura 19 – Modelo ER do Banco de Dados



Fonte: Autoria Própria (2021)

O modelo apresentado relaciona as tabelas Usuario e Tarefa. Os usuários podem estar agrupados em times, por meio do campo IDTime. Cada tarefa contém suas variáveis descritivas e é realizada por um usuário/integrante do time. Quando o integrante finaliza a realização, altera seu status para pronto e insere o atributo tempo realizado. A seguir é apresentado o dicionário de dados das tabelas do modelo.

[11] https://online.visual-paradigm.com/pt/

Quadro 6 – Dicionário de dados da tabela Usuário

Tabela	**Usuário**			
Descrição	**Armazena as informações dos usuários do sistema**			
Observações				
Campos				
Nome	**Descrição**	**Tipo de Dado**	**Tamanho**	**Restrições de Domínio (PK, FK, Not Null, Identify)**
ID	Identificação do usuário	Int		PK, Identify
Usuario	Nome de usuário no sistema	Varchar	255	Not Null
Senha	Senha de usuário	Varchar	255	Not Null
IDTime	Identificador do time em que pertence	Int		

Fonte: Autoria Própria (2021)

Quadro 7 – Dicionário de dados da tabela Tarefa

Tabela	**Tarefa**			
Descrição	**Armazena a informação das tarefas de software**			
Observações	**Essa tabela possui uma chave estrangeira da tabela Usuario**			
Campos				
Nome	**Descrição**	**Tipo de Dado**	**Tamanho**	**Restrições de Domínio (PK, FK, Not Null, Identify)**
ID	Identificação da tarefa	Int		PK, Identify
UsuarioID	Chave estrangeira referenciando o ID da tabela Usuario	Int		FK, Not Null
Descrição	Descrição da tarefa	Varchar	255	Not Null
VerboInicial	Verbo inicial da tarefa	Int		Not Null
Substantivo	Substantivo principal da tarefa	Int		Not Null
Categoria	Categoria da tarefa	Int		Not Null
Subcategoria	Subcategoria da tarefa	Int		Not Null
Status	Status da tarefa (done, doing)	Varchar	5	Not Null
TempoEstimado	Tempo estimado para a realização da tarefa	Int		
TempoRealizado	Tempo utilizado para a realização da tarefa	Int		

Fonte: Autoria Própria (2021)

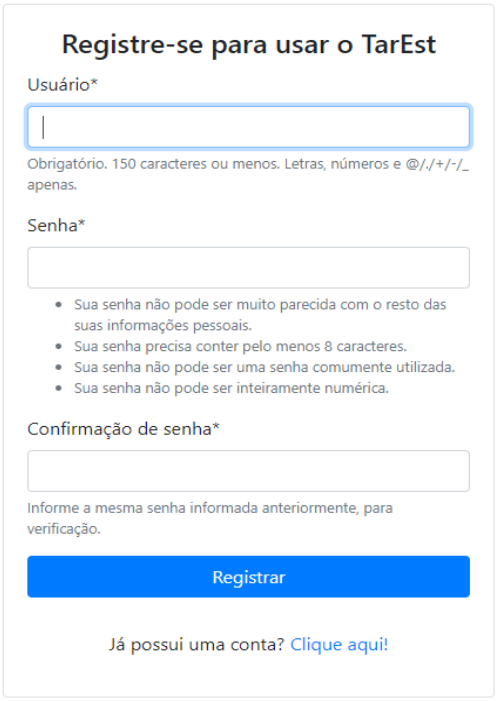
5.2.2 Implementação

Para a implementação da ferramenta optamos pelo uso do Framework Django. Inicialmente criamos as funcionalidades de cadastro, login, manter tarefas e vínculo dos membros às equipes. Num segundo momento, após criação do modelo de AM descrito na Seção 5.3, construímos a funcionalidade de Buscar Estimativa e por fim, hospedamos a aplicação para uso. Optamos pela utilização do SGBD (Sistema Gerenciados de Banco de Dados) PostgreSQL, por ser relacional, de código aberto e possuir um conjunto robusto de recursos.

Referindo-se à hospedagem da aplicação, optamos pela Heroku[12], uma plataforma gratuita, em nuvem, como um serviço que suporta várias linguagens de programação. Uma das primeiras plataformas em nuvem, o Heroku está em desenvolvimento desde junho de 2007, quando suportava apenas a linguagem de programação Ruby, mas agora suporta Java, Node.js, Scala, Clojure, Python, PHP e Go. A opção pela Heroku ocorreu especialmente devido a gratuidade e facilidade do processo de hospedagem.

A ferramenta foi nomeada TarEst[13] e seu código pode ser acessado publicamente[14]. Para utilização, o primeiro passo consiste da criação de um registro/cadastro, de modo que por meio de um usuário e senha, o usuário tenha autenticação válida e permissão de acesso. A Figura 20 apresenta a tela de registro.

Figura 20 – Tela de Registro

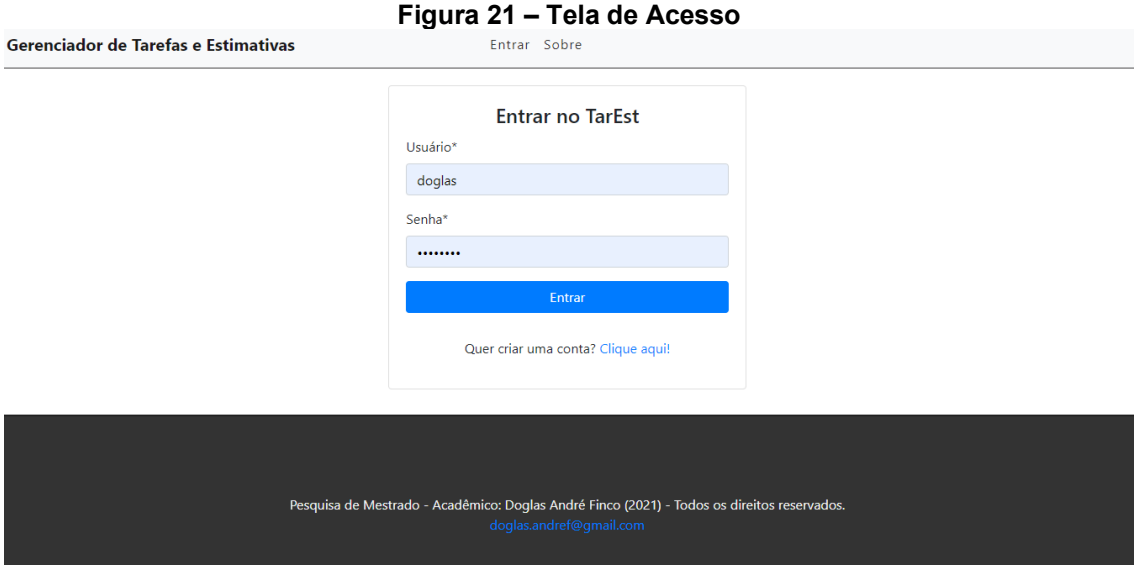


Fonte: Autoria Própria (2021)

Realizado o registro, a opção entrar (Figura 21) permite a inserção dos dados do usuário ocorrendo o acesso a aplicação em caso de credenciais válidas.

[12] https://www.heroku.com/
[13] https://tarest.herokuapp.com/
[14] https://github.com/Doglas/PesquisaMestrado/blob/master/Tarest.rar

Figura 21 – Tela de Acesso



Fonte: Autoria Própria (2021)

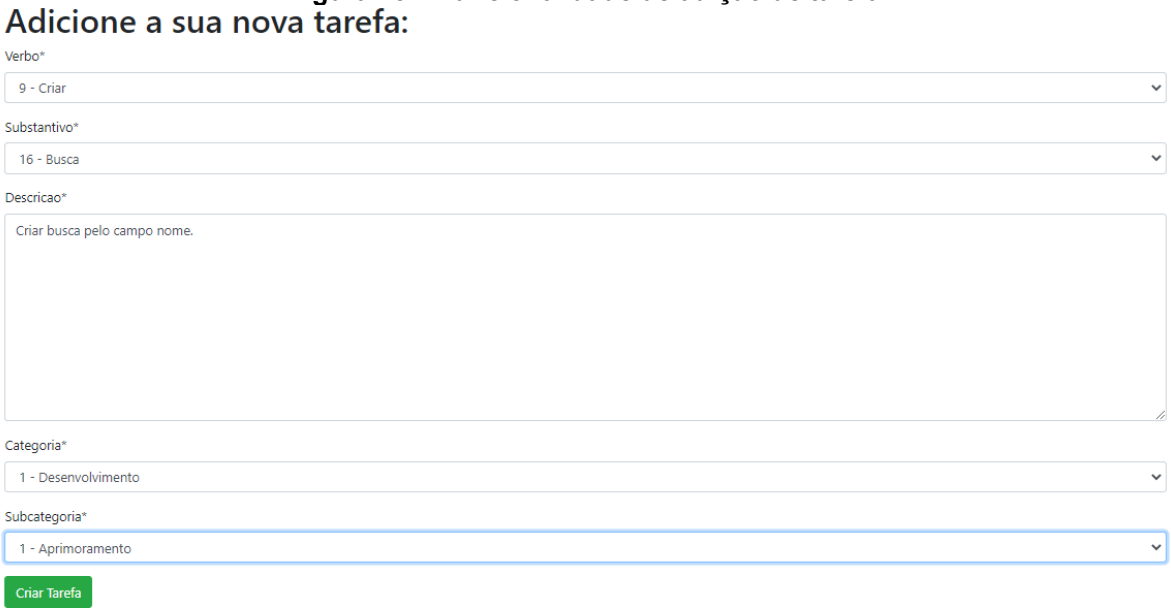
Realizado o acesso, a tela seguinte trata-se da guia Minhas Tarefas, que traz a lista das tarefas cadastradas pelo usuário (Figura 22). Além desta, outras guias estão disponíveis e ao canto superior direito aparece o usuário logado. Conforme mostrado também, aparecem as tarefas finalizadas nos últimos 30 dias, tarefas finalizadas e tarefas para fazer. As opções de adição, busca, edição/estimar e remoção de tarefas também podem ser notadas.

Figura 22 – Tela de Minhas Tarefas

Fonte: Autoria Própria (2021)

As Figuras 23 a 26 exploram as funcionalidades de adição, edição/estimativa (neste momento o foco é na edição), deleção e busca das tarefas, respectivamente.

Figura 23 – Funcionalidade de adição de tarefa



Fonte: Autoria Própria (2021)

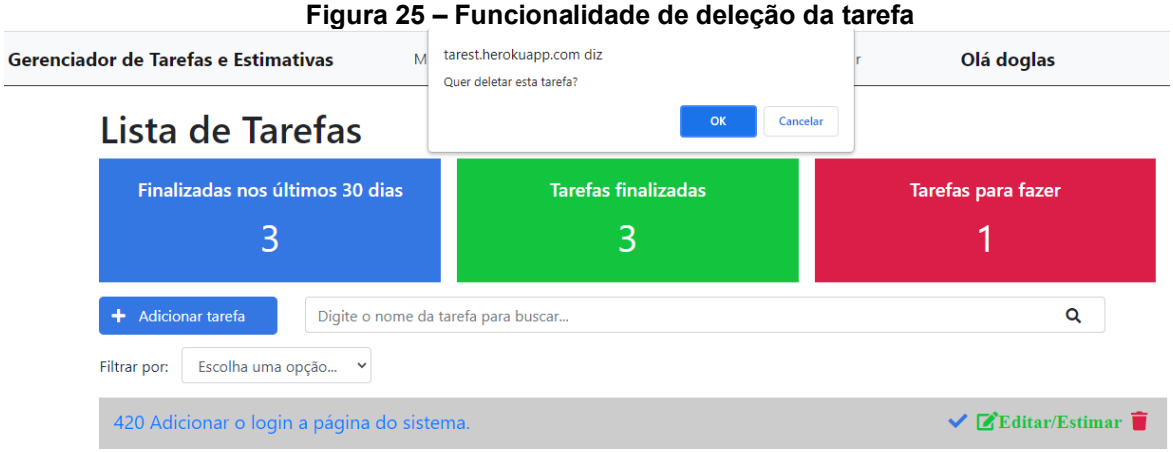
Os campos que aparecem no preenchimento das tarefas foram definidos com base no modelo de AM criado e explicado na Seção 5.3.

Figura 24 – Funcionalidade de edição de tarefa

Fonte: Autoria Própria (2021)

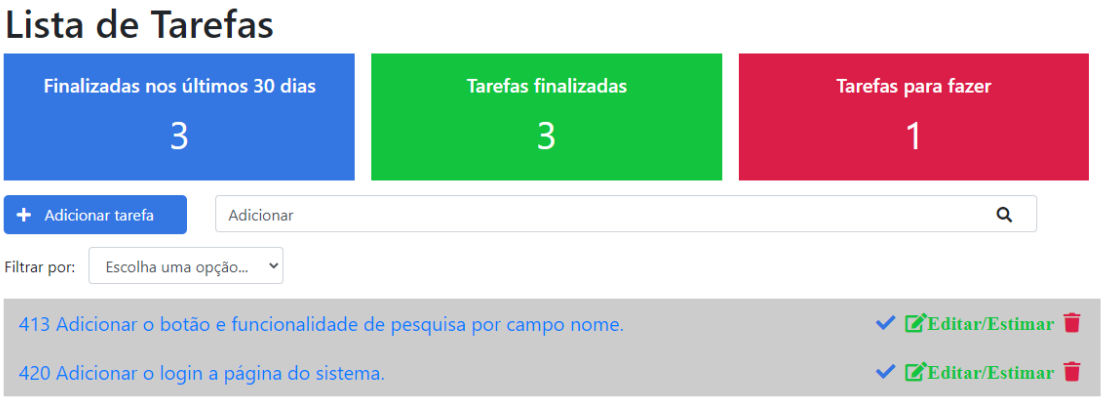
Perceba que na Tela de edição aparecem também os campos tempo estimado e tempo realizado que servem para armazenar as informações definidas no processo de estimativas.

Figura 25 – Funcionalidade de deleção da tarefa



Fonte: Autoria Própria (2021)

Figura 26 – Funcionalidade de busca



Fonte: Autoria Própria (2021)

A funcionalidade de filtro permite filtrar as tarefas por status, que podem ser: tarefas concluídas ou tarefas a fazer. As Figuras 27 e 28 exibem os filtros aplicados.

Figura 27 – Filtro de Tarefas Concluídas

Fonte: Autoria Própria (2021)

Figura 28 – Filtro de Tarefas a Fazer

Fonte: Autoria Própria (2021)

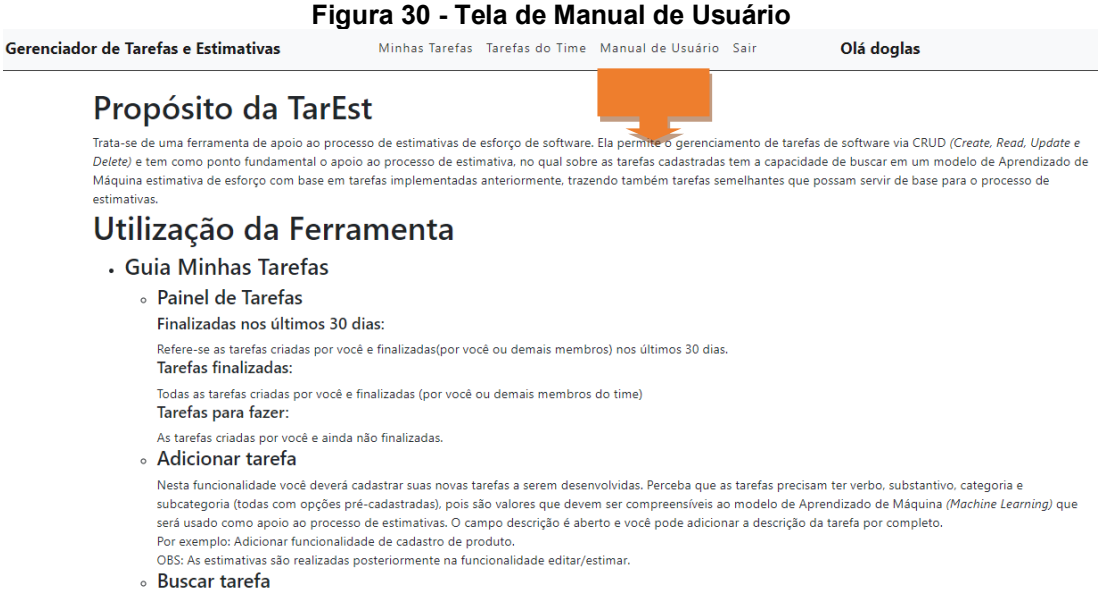
Na guia Tarefas do Time, apresentamos as tarefas que fazem parte a todos os membros da equipe. Perceba que em relação a Figura 22, a Figura 29 apresenta tarefas que foram cadastradas por outros membros da equipe. Ainda, na Lista de Tarefas do Time, não estão disponíveis as funcionalidades de cadastro ou deleção de tarefas, pois os usuários devem realizar estas operações na guia Minhas Tarefas. Isso permite que apenas os usuários que criaram as tarefas possam excluí-las.

Figura 29 – Aba Tarefas do Time

Fonte: Autoria Própria (2021)

A Funcionalidade de Manual de Usuário traz explicação quanto ao uso da ferramenta em cada funcionalidade, servindo de apoio aos utilizadores.

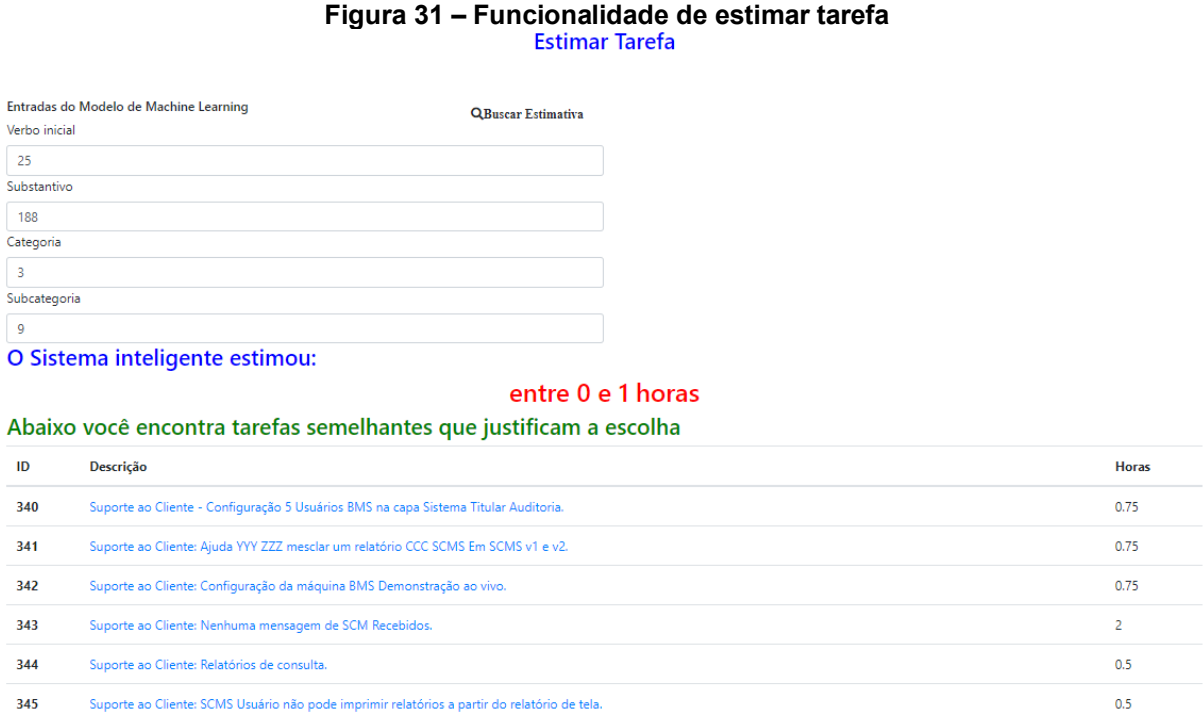
Figura 30 - Tela de Manual de Usuário



Fonte: Autoria Própria (2021)

A funcionalidade editar/estimar traz além da funcionalidade de edição, conforme a Figura 24, também aborda resultados do processo de estimativas (Figura 31). Os resultados trazidos pela funcionalidade são de acordo com o Modelo de ML construído na Seção 5.3.

Figura 31 – Funcionalidade de estimar tarefa



Fonte: Autoria Própria (2021)

Perceba que na funcionalidade de Estimar Tarefa, com base nas variáveis descritivas (verbo inicial, substantivo, categoria e subcategoria) o modelo de ML tem

a capacidade de estimar quanto tempo seria necessário para realizar a tarefa. Como forma de argumentar sobre a estimativa trazida, o modelo traz até 20 tarefas semelhantes a tarefa atual servindo de apoio a estimativa da tarefa atual.

5.3 Construção do modelo de aprendizado de máquina

O modelo de AM foi construído utilizando um *dataset* (conjunto de dados) de tarefas de softwares implementadas anteriormente e de domínio público. Conforme descrito na metodologia, seguimos as etapas do Modelo CRISP-DM para a construção do modelo, as quais são apresentadas a seguir.

5.3.1 Compreensão do negócio (Business understanding)

Quando definida a utilização do AM no processo, sabíamos da necessidade em encontrar *datasets* sobre o tema, pois sem dados não existe modelo de AM. Iniciamos as buscas visando encontrar *datasets* de tarefas de software. O que dificultava era a especificidade de ter um grande número de tarefas de software com características e tempo realizado.

Definimos uma string de busca denominada: *"dataset software task estimates"* que retornou uma base de dados com possibilidade de uso na pesquisa. O título da matéria contendo os dados chama-se: **Procurados: 99 conjuntos de dados de estimativa de esforço**[15]. O título em si define a dificuldade em encontrar *datasets* públicos com estimativas de esforço de software, realizando um apelo para que mais conjuntos de dados sejam compartilhados.

O material foi publicado em 2019 sendo composto por um arquivo contendo 10.100 estimativas de tarefas exclusivas, de dez anos de desenvolvimento comercial (entre 2004 e 2014) usando o desenvolvimento ágil. Algumas tarefas encontravam-se duplicadas devido a mais de um membro do time implementá-la, assim sendo, o arquivo inicial contava com 12299 linhas de tarefas. O conjunto de dados, nomeado de SIP[16] e algumas análises sobre os dados foram realizadas[17].

---

[15] http://shape-of-code.coding-guidelines.com/2019/01/10/wanted-99-effort-estimation-datasets/
[16] https://github.com/Derek-Jones/SiP_dataset
[17] https://arxiv.org/abs/1901.01621

5.3.2 Compreensão dos dados (Data understanding)

Encontrados os dados, estudamos as colunas existentes, buscando perceber quais poderiam ser úteis. Exploramos os tipos de dados existentes (texto, inteiros ou decimal), além de verificarmos dados duplicados, faltantes, entre outros. Após análises, decidimos utilizar as seguintes colunas para características (features) descritivas das tarefas:

- Descrição: uma breve descrição de texto da ação necessária para concluir a Tarefa, projetada para ser significativa para o cliente sempre que possível.
- Categoria: um identificador que categoriza uma Tarefa de gerenciamento, operacional ou de desenvolvimento.
- Subcategoria: um tipo específico de categoria pai, por exemplo: aprimoramento de sistema, documentação, erro, gestão, suporte, lançamento, entre outras.

Por sua vez a característica alvo (*target*) ficou definida a coluna Horas Atuais, que retorna um valor decimal equivalente ao total de horas para realizar a Tarefa.

5.3.3 Preparação dos dados (Data preparation)

Primeiramente, devido aos dados encontrarem-se em Inglês, realizamos o processo de tradução, isso se deu pois na etapa de avaliação seriam utilizados por pessoas que pudessem ter dificuldade com a língua original. Na sequência, em torno das 12299 linhas do arquivo, iniciou-se o processo de limpeza de dados. A Tabela 3 apresenta as etapas.

Tabela 3 – Processo de Limpeza dos Dados

Ação sobre o dataset	Linhas Retiradas	Linhas restantes
Dataset Inicial	-	12299
Exclusão de tarefas de Status: Canceladas, Vazio ou Autorizadas	195	12104
Exclusão de tarefas que não seguiam o padrão de iniciar com verbo, dificultando o entendimento da mesma	6943	5161
Exclusão de tarefas duplicadas (desenvolvidas por mais de um integrante), mantida apenas uma ocorrência	868	4293
Excluídas tarefas de Status: Crônicas, Estimadas e Liberadas	1360	2933

Fonte: Autoria Própria.

Após a limpeza inicial dos dados, percebemos que a coluna descrição necessitaria de processamento de linguagem natural para ser compreendida, o que aumentaria a complexidade do projeto. Deste modo, optamos em realizar o processo de engenharia de *features*, ou seja, sobre a descrição retiramos informações que pudessem contribuir com o processo de padronização das tarefas. Assim, criamos a coluna contendo o verbo inicial de cada tarefa, a partir daí o campo descrição não foi mais usado, restando o Verbo inicial, Categoria e Subcategoria como variáveis (*features*) descritivas e Horas Atuais como variável alvo (*target*).

Na sequência, analisando a característica Subcategoria, algumas não ficaram claras ou foram abordadas em situações específicas, portanto foram retiradas: Reunião de Conselho (3), Terceiros (4), Recrutamento de Pessoal (3), Pesquisa (4), Discussão (2), Inserir (2). Retirando as 18 tarefas, restaram 2915.

Posteriormente analisamos os verbos definidos, os sinônimos foram unidos e melhorias nas descrições realizadas. Por exemplo:

- Ajudar e Auxiliar foram unificados em Auxiliar;
- Apoio ao Cliente e Suporte ao Cliente foram unificados em Suporte ao Cliente;
- Desenvolver e Implementar foram unificados em Implementar;

O *dataset* ainda trazia muitas tarefas idênticas, mas com estimativas discrepantes. Então, optamos em definir o substantivo da tarefa analisando a descrição da tarefa, que na maioria das tarefas, acompanhava o verbo inicial. A criação desta nova *feature* permitiu maior diferenciação entre as tarefas. A partir dos substantivos definidos, sinônimos foram unidos e melhorias nas descrições realizadas. Por exemplo:

- Modificar e Alterar em Alterar;
- Alterar e Atualizar em Atualizar;
- Instalar e Reinstalar em Instalar;
- Preparar e Organizar em Organizar;

Em tarefas similares com estimativas discrepantes, procuramos aplicar a média, porém, os resultados não apresentaram considerável melhora, deste modo desconsideramos a média, utilizamos a mediana e pôr fim a quantidade de tarefas utilizadas ficou em 1042.

A Lista das Variáveis Descritivas é apresentada no Apêndice B e o *dataset* final pode ser encontrado publicamente[18]. As versões criadas ao longo do processo são encontradas na Tabela 4.

Para preparação dos dados, estes foram convertidos para numéricos, procedimento necessário pois os algoritmos de AM não conseguem trabalhar com texto bruto. A função *apply* da biblioteca Pandas foi utilizada para a realização do procedimento e pode ser visualizado nas Figuras 32 a 35.

Figura 32 – Conversão do Verbo Inicial para numérico

```
df['verbo_inicial'] = df['verbo_inicial'].apply(lambda x: 1 if x == 'Adicionar'
                                                else 2 if x == 'Aprender'
                                                else 3 if x == 'Atualizar'
                                                else 4 if x == 'Automatizar'
                                                else 5 if x == 'Backup'
                                                else 6 if x == 'Configurar'
                                                else 7 if x == 'Converter'
                                                else 8 if x == 'Corrigir'
                                                else 9 if x == 'Criar'
                                                else 10 if x == 'Definir'
                                                else 11 if x == 'Escrever'
                                                else 12 if x == 'Instalar'
                                                else 13 if x == 'Investigar'
                                                else 14 if x == 'Melhorar'
                                                else 15 if x == 'Mesclar'
                                                else 16 if x == 'Migrar'
                                                else 17 if x == 'Obter'
                                                else 18 if x == 'Organizar'
                                                else 19 if x == 'Permitir'
                                                else 20 if x == 'Pesquisa'
                                                else 21 if x == 'Remover'
                                                else 22 if x == 'Reunião'
                                                else 23 if x == 'Semana'
                                                else 24 if x == 'Sincronizar'
                                                else 25 if x == 'Suporte'
                                                else 26 if x == 'Testar'
                                                else 27 if x == 'Treinamento'
                                                else 28 if x == 'Validar'
                                                else 29)
```

Fonte: Autoria Própria (2021)

Figura 33 – Conversão da Categoria para numérico

```
df['categoria'] = df['categoria'].apply(lambda x: 1 if x == 'Desenvolvimento'
                                        else 2 if x == 'Gestão'
                                        else 3)
```

Fonte: Autoria Própria (2021)

[18] https://github.com/Doglas/PesquisaMestrado/blob/master/CSVTarefas.csv

Figura 34 - Conversão da Subcategoria para numérico

```
df['subcategoria']=df['subcategoria'].apply(lambda x: 1 if x == 'Aprimoramento'
                                            else 2 if x == 'Consultoria'
                                            else 3 if x == 'Documentação'
                                            else 4 if x == 'Erro'
                                            else 5 if x == 'Lançamento'
                                            else 6 if x == 'Marketing'
                                            else 7 if x == 'Reunião de Gestão'
                                            else 8 if x == 'Reunião de progresso'
                                            else 9 if x == 'Suporte ao Cliente'
                                            else 10 if x == 'Suporte Interno'
                                            else 11 if x == 'Teste'
                                            else 12)
```

Fonte: Autoria Própria (2021)

Para a transformação do substantivo, devido à grande quantidade de diferentes substantivos, utilizamos a função *OrdinalEncoder* que codifica valores categóricos em um vetor de inteiros. A Figura 35 apresenta a transformação.

Figura 35 - Conversão do Substantivo para numérico

```
from sklearn.preprocessing import OrdinalEncoder

enc = OrdinalEncoder()
X = df[['substantivo']]
enc.fit(X)
OrdinalEncoder()
df[['substantivo']] = enc.transform(df[['substantivo']])
```

Fonte: Autoria Própria (2021)

Quanto à utilização da variável alvo, as Horas Atuais (quantidade de horas para realizar a tarefa), inicialmente pretendíamos utilizar as cartas do *Planning Poker* para classificação. Porém, ao longo dos experimentos, percebemos dificuldade na escolha de uma carta específica, assim, optamos em criar categorias para a variável alvo, ou seja, agrupamos as cartas definindo intervalos.

A conversão da variável alvo ocorreu com base na quantidade de ocorrências da base de dados, sendo algumas variações dos intervalos ocorreram, até chegarmos nas categorias da Figura 36.

Figura 36 - Conversão da variável alvo

```
def categorizaTarget(s):
    if s <= 1.5:
       return int(0)
    elif s > 1.5 and s <= 4.5:
       return int(1)
    elif s > 4.5 and s <= 9.5:
       return int(2)
    elif s > 9.5:
       return int(3)
```

```
df['horas'] = df['horas'].apply(categorizaTarget)
```

Fonte: Autoria Própria (2021)

Após o pré-processamento dos dados, a Figura 37 apresenta as primeiras linhas da base de dados através da função *head.*

Figura 37 - Dados após o processo de conversão

```
df.head()
```

	verbo_inicial	substantivo	categoria	subcategoria	horas
0	1	12.0	3	10	0
1	1	12.0	1	1	1
2	1	15.0	1	4	0
3	1	15.0	1	1	1
4	1	15.0	1	1	1

Fonte: Autoria Própria (2021)

Na sequência, definimos as variáveis descritivas e variável alvo (Figura 38).

Figura 38 - Variáveis descritivas e variável alvo

```
y = df[['horas']] # variável alvo (target)
X = df.iloc[:, 0:4] # variáveis descritivas (features)
```

Fonte: Autoria Própria (2021)

Após a preparação dos dados, o próximo passo consiste em dividir dados de teste e treino para posterior aplicação dos algoritmos. Utilizamos a função *train_test_split* para realizar a divisão, seguindo a proporção 70% para treino dos modelos e 30% para teste, divisão comumente utilizada.

Figura 39 - Divisão de dados (teste e treino)

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0, stratify=y)
```

Fonte: Autoria Própria (2021)

5.3.4 Modelagem (Modeling)

Realizada a preparação dos dados, definimos algoritmos consolidados de AM para aplicação. Dentre tantos, optamos por um modelo baseado em informação, a Árvore de Decisão; um em similaridade *k-NeighborsClassifier*; um em probabilidade, o *Naive Bayes;* e por fim o XGBoost, um modelo baseado em árvore de decisão e que utiliza uma estrutura de *Gradient boosting*.

Ao utilizarmos os algoritmos não modificamos nenhum parâmetro, ou seja, utilizamos os valores padrões de cada modelo. A chamada de cada algoritmo é apresentada na Figura 40.

Figura 40 – Chamada das Funções de Cada Modelo

```
#Chamando a função da Árvore de Decisão
dectree = tree.DecisionTreeClassifier()
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

```
#Chamando a função do KNN
knn = KNeighborsClassifier()
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

```
#Chamando a função de Naive Bayes
naiveb = GaussianNB()
```

```
GaussianNB(priors=None, var_smoothing=1e-09)
```

```
#Chamando a função do XGBoost
xgb = XGBClassifier()
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0,
              learning_rate=0.1, max_delta_step=0, max_depth=3,
              min_child_weight=1, missing=None, n_estimators=100, n_jobs=1,
              nthread=None, objective='multi:softprob', random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
              silent=None, subsample=1, verbosity=1)
```

Fonte: Autoria Própria (2021)

5.3.5 Avaliação (Evaluation)

A avaliação analisa o desempenho dos modelos construídos. Utilizamos a avaliação via *classification report,* que mostra uma representação das principais métricas de classificação por classe e permite uma análise mais profunda do comportamento do classificador sobre a precisão global (SCIKIT-YB, 201-). De acordo com a Scikit-yb (201-), as métricas do *classification report* são:

- *precision*: pode ser vista como uma medida da exatidão de um classificador. Para cada classe, é definido como a proporção de verdadeiros positivos para a soma de verdadeiros e falsos positivos. Dito de outra forma, "para todas as instâncias classificadas como positivas, qual porcentagem estava correta?";
- *recall*: é uma medida da integridade do classificador; a capacidade de um classificador de encontrar corretamente todas as instâncias positivas. Para cada classe, é definido como a proporção de verdadeiros positivos para a soma de verdadeiros positivos e falsos negativos. Dito de outra forma, "para todas as instâncias que foram realmente positivas, qual porcentagem foi classificada corretamente?";
- *f1-score*: é uma média harmônica ponderada de *precision* e *recall*, de modo que a melhor pontuação é 1,0 e a pior é 0,0. De modo geral, as pontuações f1 são mais baixas do que as medidas de *precision*, pois incorporam *precision* e *recall* em seus cálculos. Como regra geral, a média ponderada de f1 deve ser usada para comparar os modelos do classificador, não a precisão global;
- *support*: é o número de ocorrências reais da classe no conjunto de dados. O suporte desequilibrado nos dados de treinamento pode indicar fraquezas estruturais nas pontuações relatadas do classificador e indicar a necessidade de amostragem estratificada ou rebalanceamento. O suporte não muda entre os modelos, mas diagnostica o processo de avaliação.

O nosso foco sobre o *classification report* foi na métrica *weighted avg*, da coluna *f1-score* por atuar sobre cada rótulo, encontrando sua média ponderada por suporte (o número de instâncias verdadeiras para cada rótulo). Esta métrica compensa o desequilíbrio da quantidade de dados por rótulo, servindo de indicador adequado para as análises. Os resultados da métrica aplicada sobre as várias versões dos modelos encontram-se na Tabela 4.

Tabela 4 - Processo de Avaliação dos Modelos

VERSÃO	SITUAÇÃO DE AVALIAÇÃO	RESULTADOS
V1	Features descritivas utilizadas (verbo_inicial, categoria e subcategoria). Target como cartas do Planning Poker.	• Árvore de Decisão: 0,30; • k-NN: 0,29; • Naive Bayes: 0,19; • XGBoost: 0,28
V2	Features descritivas com adição do substantivo (verbo_inicial, substantivo, categoria e subcategoria). Target como cartas do Planning Poker.	• Árvore de Decisão: 0,29; • k-NN: 0,30; • Naive Bayes: 0,18; • XGBoost: 0,32
V3	Features descritivas da versão V2. Agrupamento das cartas 13, 20, 40,100 a categoria 13 horas ou mais.	• Árvore de Decisão: 0,33; • k-NN: 0,33; • Naive Bayes: 0,21; • XGBoost: 0,35;
V4	Com base no agrupamento de cartas da V3, aplicada média nas tarefas similares ainda com discrepância na variável target.	• Árvore de Decisão: 0,29; • k-NN: 0,29; • Naive Bayes: 0,27; • XGBoost: 0,32;
V5	Com base na V3 aplicada e mediana em tarefas similares com discrepância e categorização da variável alvo para intervalos: [0,1], [2,4], [5,7], [8, 10], [>10].	• Árvore de Decisão: 0,36; • k-NN: 0,39; • Naive Bayes: 0,28; • XGBoost: 0,45;
V6	Com base na V3 e mediana em tarefas similares com discrepância e categorização da variável alvo para intervalos: [0,1], [2,4], [5,9], [>9].	• Árvore de Decisão: 0,41; • k-NN: 0,39; • Naive Bayes: 0,31; • XGBoost: 0,46;
V7	Após melhorias da categorização dos substantivos, com base na V6.	• Árvore de Decisão: 0,54; • k-NN: 0,5; • Naive Bayes: 0,41; • XGBoost: 0,53;

Fonte: Autoria Própria (2021)

A Tabela 4 apresentou os modelos de AM construídos ao longo do processo. As versões V1 até V6 não trouxeram bons resultados e a V7 apresentou resultados utilizáveis, com qualidade razoável. Sobre ela utilizamos o algoritmo Árvore de Decisão e a Figura 41 exibe o *classification_report* do modelo.

Figura 41 - Resultados do Modelo

```
print(classification_report(y_test, previsoesdectree))
```

	precision	recall	f1-score	support
0	0.62	0.63	0.62	110
1	0.56	0.55	0.56	111
2	0.45	0.47	0.46	53
3	0.41	0.38	0.39	39
accuracy			0.54	313
macro avg	0.51	0.51	0.51	313
weighted avg	0.54	0.54	0.54	313

Fonte: Autoria Própria (2021)

Os resultados do *classification_report* indicam para um desempenho razoável do modelo, considerando a *weighted avg* da *f1-score* em 0,54. Quanto mais próximo de 1 é o desempenho, melhor a acurácia do modelo, o cenário ideal.

Em torno da qualidade razoável do modelo, percebemos situações no *dataset* que poderiam resultar em problemas. Dentre eles, destacamos que se trata da exploração de dados num período de 10 anos, sobre vários projetos de software, que contaram com 22 membros diferentes. Por referir-se aos anos 2004 e 2014 a realidade da época não exigia uma padronização tão grande das tarefas. O ato de ter poucas colunas (variáveis descritivas) para diferenciar as tarefas dificultou o aprendizado dos modelos, exigindo a criação de *features* para que resultados aceitáveis fossem atingidos. O modelo completo com as etapas apresentadas e descritas está disponível para acesso[19].

Sobre o modelo de AM fica o chamamento para outros *datasets* sejam compartilhados, permitindo a construção de modelos mais assertivos e consequentemente melhores resultados alcançados.

5.3.6 Implantação (Deployment)

O processo de implantação se deu inserindo o modelo criado na funcionalidade Estimar Esforço da ferramenta TarEst, apresentada na Figura 31. A nível de código a essência da chamada ao modelo é apresentado na Figura 42.

Figura 42 - Chamada do Modelo na Ferramenta

```
temp={}
temp['a']=request.GET.get('search')
temp['b']=request.GET.get('substantivo')
temp['c']=request.GET.get('categoria')
temp['d']=request.GET.get('subcategoria')
testDtaa=pd.DataFrame({'x':temp}).transpose()
horas=reloadModel.predict(testDtaa)[0]
```

Fonte: Autoria Própria (2021)

Conforme o código anterior, trata-se de enviar as variáveis descritivas da tarefa ao modelo, e posteriormente o modelo retorne à categoria de horas estimadas. A categoria de horas conforme definido na Figura 36, refere-se a um

[19] https://github.com/Doglas/PesquisaMestrado/blob/master/MLMestradoProducao.ipynb

intervalo de horas e para que o intervalo de horas fosse visualizado corretamente, ajustes ocorreram conforme Figura 43.

Figura 43 - Retorno da categoria horas pelo modelo

```
<div class="offset-1 col-10 list-div">
    {% if horas == 0 %}
        <font color="blue"><h3>O Sistema estimou:</h3></font>
        <center><font color="red", ><h2>entre 0 e 1 horas</h2></font></center>
        <font color="green"><h3>Abaixo você encontra tarefas semelhantes que justificam a escolha</h3></font>
    {% elif horas == 1 %}
        <font color="blue"><h3>O Sistema estimou:</h3></font>
        <center><font color="red", ><h2>entre 2 e 4 horas</h2></font></center>
        <font color="green"><h3>Abaixo você encontra tarefas semelhantes que justificam a escolha</h3></font>
    {% elif horas == 2 %}
        <font color="blue"><h3>O Sistema estimou:</h3></font>
        <center><font color="red", ><h2>entre 5 e 9 horas</h2></font></center>
        <font color="green"><h3>Abaixo você encontra tarefas semelhantes que justificam a escolha</h3></font>
    {% elif horas == 3 %}
        <font color="blue"><h3>O Sistema estimou:</h3></font>
        <center><font color="red", ><h2>mais do que 9 horas</h2></font></center>
        <font color="green"><h3>Abaixo você encontra tarefas semelhantes que justificam a escolha</h3></font>
    {% endif %}
</div>
```

Fonte: Autoria Própria (2021)

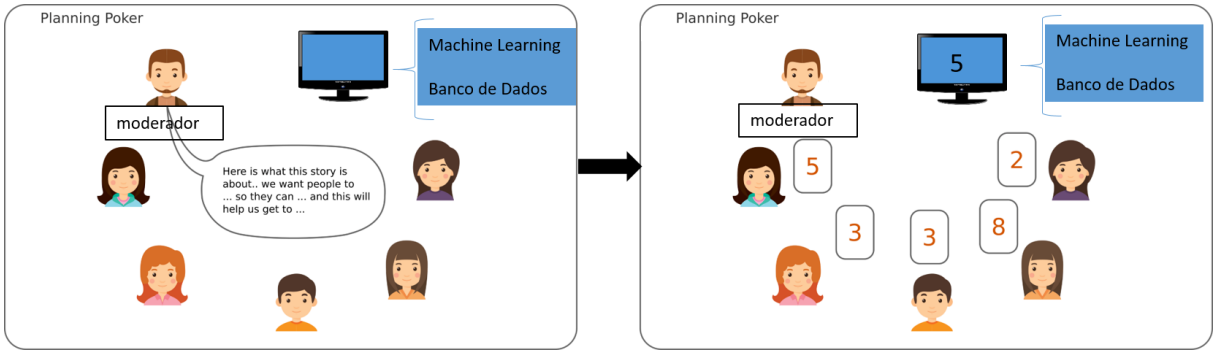
5.4 Documentação da proposta

Esta Seção mostra a inserção da *ML Planning Poker* na prática e como ocorre o processo. Conforme representado na Figura 44(a), a equipe se prepara para o processo de estimativas das tarefas da *sprint*, com participação do moderador (podendo ser membro da equipe), equipe de desenvolvimento e a ferramenta desenvolvida, agindo similarmente a um integrante do time.

A equipe recebe a tarefa a ser estimada e descrita pelo moderador. Cada membro define sua carta do *Planning Poker* e o moderador insere a tarefa na ferramenta. Todos os membros apresentam suas estimativas simultaneamente, inclusive a ferramenta, conforme Figura 44(b). A ferramenta traz complementarmente a sua estimativa, uma lista com tarefas semelhantes realizadas em estimativas anteriores (Figura 31), como justificativa à escolha. Perante o debate sobre as estimativas levantadas a ferramenta não conseguirá alterar sua escolha, pois não é capaz de mudar de opinião via debate, mas os membros da equipe continuam esclarecendo sobre a tarefa até finalizar o processo do *Planning Poker* e definição da estimativa. O processo se repete para toda tarefa a ser estimada.

Figura 44 – Esboço do projeto na prática

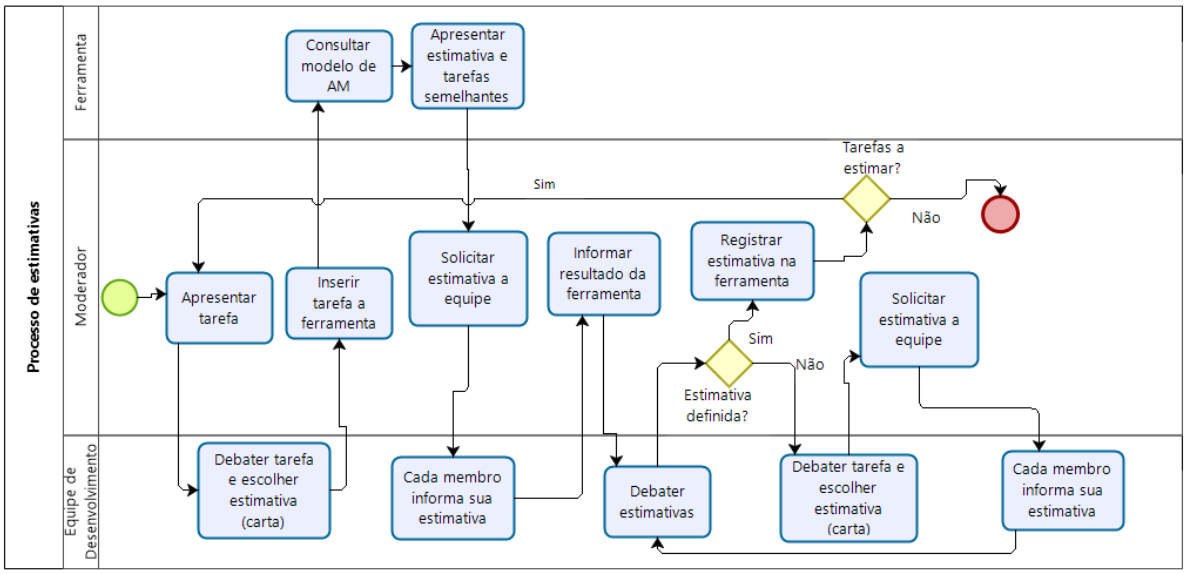
(a) Descrição da tarefa pelo moderador. (b) Estimativa simultânea dos membros.



Fonte: Autoria Própria (2021). Baseado em: (HICKS, 2020)

A Figura 45 apresenta o diagrama do processo de estimativas da *ML Planning Poker* através da ferramenta Bizagi[20] que segue o padrão BPMN (*Business Process Model and Notation*) e por meio das raias procurando facilitar a visualização do responsável por realizar cada ação no processo.

Figura 45 – Etapas da proposta



Fonte: Autoria Própria (2021)

O processo ocorre da seguinte maneira:

1. Inicia-se o processo de estimativas das tarefas da Sprint atual. Uma nova tarefa é apresentada pelo moderador aos integrantes da equipe de desenvolvimento. O moderador responde todas as perguntas que os membros da equipe têm relacionado à tarefa;

[20] https://www.bizagi.com/pt/plataforma/modeler

2. Após as perguntas serem respondidas, cada estimador seleciona de maneira privada uma das cartas que representa sua estimativa. A carta escolhida é baseada no conhecimento do estimador. As cartas não são mostradas até que todos os membros tenham feito suas escolhas;
3. O moderador insere a tarefa na ferramenta e após a funcionalidade de Estimar Tarefa (Figura 31) é utilizada para que traga a estimativa para a tarefa, bem como tarefas semelhantes implementadas anteriormente;
4. A partir do momento em que todo membro realizou sua escolha, o moderador solicita que as cartas sejam viradas simultaneamente para que todos possam ver cada estimativa;
5. Na sequência, o moderador mostra a estimativa definida pela ferramenta e as tarefas semelhantes desenvolvidas anteriormente;
6. Nesta etapa, os resultados podem divergir significativamente. Se isso ocorrer, os estimadores que escolheram os valores mais alto e mais baixo explicam os fatores analisados e os motivos para a escolha daquela estimativa. A estimativa trazida pela ferramenta é debatida, bem como as situações que geraram dúvidas;
7. Se após esta rodada a equipe chegar ao consenso ou a uma definição da estimativa para a tarefa, a estimativa é inserida na ferramenta pelo moderador no campo Tempo Estimado. Havendo outra tarefa a ser estimada, retorna a etapa 1 do processo;
8. Se não ocorrer a definição da estimativa, mais uma vez cada estimador seleciona de maneira privada uma das cartas que representa sua estimativa. As cartas não são mostradas até que todos os membros tenham feito suas escolhas e a partir do momento em que todo membro realizou sua escolha, o moderador solicita que as cartas sejam viradas simultaneamente para que todos possam ver cada estimativa. Ocorre novamente a etapa 6 do processo. É importante ressaltar que a ferramenta não irá alterar sua estimativa, haja vista que, não tem a capacidade de mudar de opinião via debate. A repetição das etapas 6, 7 e 8 ocorrem até que atinja o consenso ou razoabilidade de estimativa de cada tarefa.

Percebemos na proposta que o modelo de AM apoia o processo de estimativas, trazendo tarefas realizadas anteriormente como base para estimativa da tarefa atual, ressaltando que a técnica *Planning Poker* continua ocorrendo, uma vez

que o fator humano é fundamental ao processo. Note que o informe da estimativa definida pela ferramenta ocorre posteriormente ao informe da estimativa dos participantes, isso se dá buscando evitar que os estimadores sejam influenciados pelos resultados da ferramenta.

Defendemos a combinação perante o contexto em que a troca de conhecimento entre os integrantes da equipe se mantenha e os resultados trazidos através do *Planning Poker* continuem sendo valorizados. Mas também, o AM também seja consultado como um artefato complementar ao processo, gerando assim uma combinação que obtenha o melhor do fator humano juntamente a análise de dados históricos. Quando ocorre disparidade entre os resultados do *Planning Poker* e AM é por meio do debate e rodadas posteriores do *Planning Poker* que se define a estimativa.

5.5 Avaliação

Conforme definido na Seção 4.5, a avaliação ocorreu de maneira qualitativa e quantitativa. A dificuldade de inserção nas empresas impediu a avaliação em ambiente real. Deste modo, realizamos avaliação em sala de aula com estudantes de graduação e com profissionais de TI.

5.5.1 Ambiente Acadêmico

Para a avaliação em ambiente acadêmico utilizamos duas turmas de estudantes na disciplina de Projeto e Análise de Sistemas nos cursos de graduação de Sistemas da Informação e Engenharia da Computação da UTFPR. Participaram da pesquisa 32 estudantes divididos em 9 grupos. Cada grupo definiu um sistema para implementar e tinha o desafio de organizar-se para o desenvolvimento da aplicação, dentre as aplicações desenvolvidas podemos citar: sistema para gestão de restaurante, sistema para manutenção de funcionários e sistema de gestão para postos de lavação de veículos. Todos os grupos elencaram um líder que representava a equipe e tirava as dúvidas referente aos procedimentos, servindo de elo aos pesquisadores. O processo de estimativas ocorreu na aplicação final da disciplina, em que cada equipe definiu um sistema e necessitava implementá-lo.

Realizamos a abordagem antes/depois. Inicialmente aplicado o *Planning Poker* original na metade das tarefas estimadas e posteriormente estimada a outra metade das tarefas utilizando a *ML Planning Poker*, ao final da implementação das tarefas os membros da equipe armazenaram também seu tempo realizado.

O pesquisador participou e interveio em algumas aulas da disciplina, explicando os procedimentos necessários para atingir os objetivos de avaliação. Os passos seguidos e seus resultados encontram-se na Tabela 5.

1. Os estudantes conheceram o funcionamento do *Planning Poker* através do professor titular da disciplina;
2. Ocorreu um contato inicial do pesquisador com os estudantes em que foi solicitado às equipes que definissem as tarefas de software da aplicação e estimassem metade das mesmas utilizando o *Planning Poker* original;
3. Duas semanas depois, o pesquisador retornou para apresentar a *ML Planning Poker*, solicitando que o restante das tarefas (outra metade) fosse estimado usando a proposta;
4. Realizado o processo de estimativa de todas as tarefas definidas, solicitamos aos estudantes que armazenassem também o tempo gasto para realizar as tarefas, permitindo análises posteriores.

Tabela 5 – Abordagem antes/depois

ANTES/DEPOIS (*PLANNING POKER* ORIGINAL VERSUS *ML PLANNING POKER*)

	Planning Poker original		*ML Planning Poker*	
TAREFAS	37	100%	37	100%
ASSERTIVIDADE	9	24,3%	9	24,3%
SUBESTIMATIVA	7	18,9%	13	35,1%
SUPERESTIMATIVA	21	56,7%	15	40,5%
TAREFAS SUBESTIMADAS OU SUPERESTIMADAS				
TAREFAS	28	100%	28	100%
DIFERENÇA <=1	11	39,2%	16	57,1%
DIFERENÇA > 1	17	60,7%	12	42,8%

Fonte: Autoria Própria (2021)

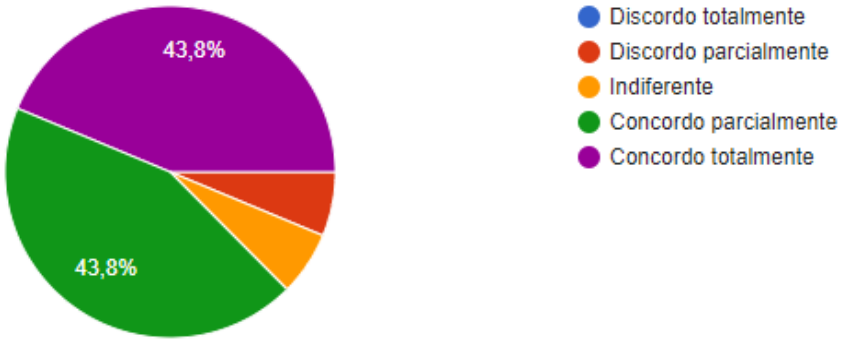
Na comparação apresentada, visualizamos a assertividade (igualdade no estimado *versus* realizado) em 24,3% em ambas as propostas. O *Planning Poker* original teve um número maior de tarefas superestimadas (56,75%) em relação ao total, por sua vez, a *ML Planning Poker* teve um equilíbrio entre tarefas subestimadas (35,1%) e superestimadas (40,5%). Analisando as tarefas com

estimativa incorreta, percebemos que a proposta teve melhor resultado, considerando-se que, 57,1% das tarefas teve diferença máxima de 1 hora entre estimado e realizado em comparação a 39,2% do *Planning Poker* original. Deste modo também, 60,7% das tarefas do *Planning Poker* original teve erro maior de 1 hora em comparação a 42,8% da *ML Planning Poker*. As tarefas utilizadas nas análises da Tabela 5 encontram-se no Apêndice C.

Após a realização do processo com a *ML Planning Poker*, aplicamos um questionário com os participantes com intuito de compreender as percepções obtidas. O formulário foi respondido de modo remoto (via Google Forms) e encontra-se no Apêndice A. Os participantes receberam a explicação do estudo por meio de um termo livre e esclarecido (Apêndice D), optando em participar ou não da pesquisa. A Q1 refere-se a saber se o participante é alguém que utilizou a ferramenta e tem conta válida, visando garantir a veracidade sobre os resultados obtidos.

O Gráfico 1 apresenta os resultados da Q2, que explorou o seguinte: “Qual é sua percepção quanto a afirmação: O debate ocorrido no *Planning Poker* é auxiliado com o uso da proposta”.

Gráfico 1 – Resultado da Q2 (Acadêmicos)

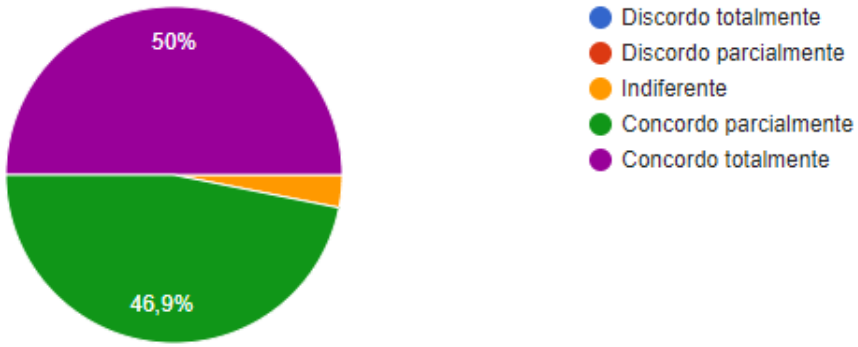


Fonte: Autoria Própria (2021)

A maioria dos respondentes (87,6%) concordam (parcialmente ou totalmente) que o debate ocorrido no *Planning Poker* é auxiliado. Isso pode ocorrer pois os estimadores sem experiência com a implementação da tarefa, tem a possibilidade de analisar tarefas implementadas anteriormente e com base nestas análises consegue debater junto ao processo.

Na Q3, temos: “Qual é sua percepção quanto a afirmação: O *Planning Poker* mantém suas características essenciais com o uso da proposta”. O Gráfico 2 apresenta os resultados.

Gráfico 2 – Resultado da Q3 (Acadêmicos)

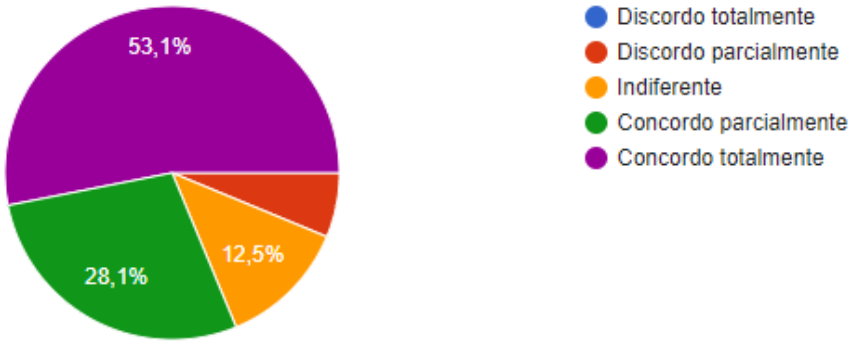


Fonte: Autoria Própria (2021)

A percepção de 96,9% dos participantes (que concordam totalmente ou parcialmente) é que o *Planning Poker* mantém suas características essenciais, isso pode condizer com o fato da ferramenta atuar como apoio ao processo, sem ignorar os aspectos de debate e ciclos realizados na técnica original. Fortalecemos que o *Planning Poker* permite uma maneira dinâmica e agradável para estimar, sendo assim, sempre houve preocupação para que a essência da prática prevalecesse.

No que se refere a Q4: “Qual é sua percepção quanto a afirmação: A combinação do *Planning Poker* com a proposta contribui no processo de estimativas.”, buscamos identificar se os participantes percebessem contribuição da proposta na melhoria do processo de estimativas, uma das questões mais importantes de nossa pesquisa, os resultados encontram-se no Gráfico 3.

Gráfico 3 – Resultado da Q4 (Acadêmicos)



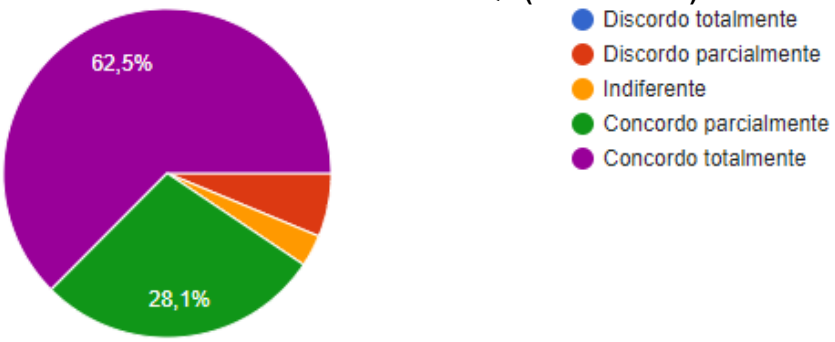
Fonte: Autoria Própria (2021)

Percebemos que 81,2% acreditam na contribuição da *ML Planning Poker*, destes 53,1% concordam totalmente, enquanto 28,1% concordam parcialmente. Isso

nos permite acreditar no potencial da proposta, trazendo uma percepção positiva quanto ao uso. Esta resposta vai ao encontro ao objetivo geral do trabalho, direcionando para uma interferência positiva da combinação das técnicas na melhoria do processo de estimativas de esforço de software.

No que diz respeito a Q5: “Qual é sua percepção quanto a afirmação: A proposta é fácil de ser compreendida.”, os resultados estão no Gráfico 4.

Gráfico 4 - Resultado da Q5 (Acadêmicos)

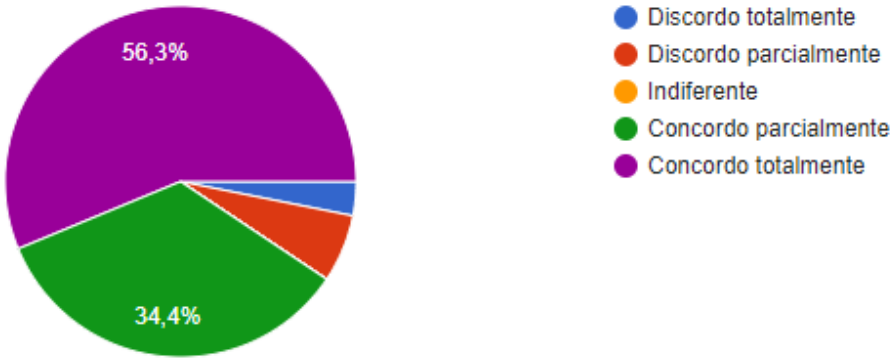


Fonte: Autoria Própria (2021)

As respostas colhidas direcionam para a facilidade de compreensão da proposta 90,6% referentes a concordância parcial ou total da afirmação. Isso vai de encontro os princípios ágeis, facilitando a compreensão dos processos, sem exigir tempo excessivo aos utilizadores.

No que diz respeito a Q6: “Qual é sua percepção quanto a afirmação: A proposta é fácil de ser utilizada.”, temos os seguintes dados (Gráfico 5).

Gráfico 5 - Resultado da Q6 (Acadêmicos)



Fonte: Autoria Própria (2021)

As percepções apontam para a facilidade de utilização da proposta com 91,7% dos resultados. A facilidade aumenta as chances de uso por parte das equipes, uma vez que não torna o processo complexo ou engessado.

No que diz respeito a Q7: “Quais são as percepções positivas quanto ao uso da proposta?”. Algumas das respostas são apresentadas no Quadro 8.

Quadro 8 – Percepções positivas a proposta (Acadêmicos)

Resposta
Gostei da utilização da ferramenta, tanto em relação à interface por ser fácil de usar e muito intuitiva, quanto na hora de interpretar os resultados.
Achei muito legal a proposta, com certeza contribui no dia-a-dia do desenvolvedor e em suas estimativas que ficarão mais próximas da realidade.
Estimativa de tempo da ferramenta auxiliou bastante minha equipe com tarefas que não possuímos experiência para serem avaliadas.
Por se tratar de uma estimativa adicional à realizada com o planning poker, traz uma segurança maior em relação ao tempo gasto com cada etapa estimada.
A proposta auxilia muito a discussão por trazer com base tarefas que já foram realizadas e quanto tempo elas utilizaram. Há situações onde os membros podem não ter conhecimento suficiente para estimar, mas a proposta mostra várias tarefas que a equipe pode ler e analisar qual é mais parecida com aquela que querem estimar.
A proposta oferece exemplos de tarefas e quanto tempo elas levaram, e ter essas informações para consultar ajuda na estimativa
Acredito que é muito interessante fazer o uso do Planning Poker e Aprendizado de Máquina, pois facilita muito mais modelar e entender um problema que será resolvido em uma equipe.
Acredito que a estimativa feita pela ferramenta ajuda a ter uma base melhor para discussões, já que há algumas tarefas que são mais difíceis de se estimar por conta.
Com um ótimo dataset, leva a reduzir o tempo de debate, devido muitos projetos terem muitas tarefas repetidas. Além disso, pode auxiliar para um consenso da equipe.

Fonte: Autoria Própria (2021)

As percepções dos participantes direcionam para pontos positivos nas tarefas em que os membros não possuem experiência, haja vista que a proposta traz atividades similares realizadas, trazendo segurança maior ao participante na definição de sua estimativa. Além disso, estes exemplos similares podem reduzir o tempo de debate agilizando a definição da estimativa.

No que diz respeito a Q8: “Quais são as percepções negativas quanto ao uso da proposta?”, algumas das respostas compiladas estão no Quadro 9.

Quadro 9 – Percepções negativas a proposta (Acadêmicos)

Resposta
Meu grupo ficou um pouco confuso em relação a qual verbo utilizar para cada atividade e como descrever as atividades de forma correta.
A interface do usuário pode ter melhorias em relação à experiência de uso, repensar um pouco os botões e a maneira que as informações são apresentadas.
Tive um pouco de dificuldade para enquadrar as minhas tarefas dentro das opções disponíveis na ferramenta. Alguns tópicos poderiam ter uma descrição mais detalhada sobre o que representam.
Encontramos um intervalo meio grande de tempo em algumas tarefas que estávamos estimando na plataforma, nessas acredito que a plataforma não nos ajudou tanto na estimativa.
O fluxo para gerar a estimativa de tempo foi um pouco confuso no primeiro uso. Poderia ser uma informação já disposta ao criar uma tarefa.
Na parte de criar a tarefa há vários substantivos, porém creio que há poucos verbos. O que faz com que a estimativa a inteligência artificial use com base tarefas que demandam muito tempo, e a média pode ficar superestimada. Talvez seria interessante também permitir mais de um substantivo por tarefa.
Em alguns casos não concordamos com o tempo necessário sugerido pela proposta.
Acredito que a única percepção negativa que eu vejo é que no começo a equipe pode estimar da forma errada por falta de experiência com essas técnicas, mas isso se corrige com o tempo.

Achei um tanto confusa a categorização das tarefas, houve algumas que foram difíceis de representar com exatidão nos termos disponibilizados.
Como a proposta tem uma base de dados aparentemente pequena, acaba cometendo alguns deslizes, com tempos muito pequenos ou muito grandes, o que atrapalha um pouco na organização e decisão de como e quando executar o projeto.

Fonte: Autoria Própria (2021)

Percebe-se que algumas questões geraram dúvida e sugestões foram apresentadas, dentre elas, os participantes destacaram os campos fixos que devem ser inseridos na tarefa, referindo-se as variáveis descritivas do modelo de ML e acabam sendo engessados, tal que os usuários fiquem presos as características conhecidas pelo modelo.

Referente ao relato da pequena quantidade de verbos, é importante ressaltar que se fez necessário unir verbos sinônimos pois representavam a mesma situação e anteriormente quando eram percebidos como tarefas diferentes atrapalhavam a classificação. Deste modo, foi necessário reduzir o número de verbos para que um modelo de melhor acurácia fosse obtido.

Outra situação, diz respeito a tarefas com estimativas pouco úteis trazidas pela ferramenta, isso ocorre devido ao *dataset* ter sido construído ao longo de 10 anos, em tempos que não era exigida uma padronização no cadastro das tarefas. Além disso, as tarefas realizadas por outros profissionais não trazer a realidade dos membros que estão estimando atualmente, portanto, num cenário ideal, as tarefas armazenadas na ferramenta deveriam ter sido implementadas pela equipe que está estimando o que tornaria o cenário mais próxima da realidade da equipe.

No que diz respeito a Q9: “Qual a sua percepção quanto a combinação de técnicas para estimativa de esforço de software (neste caso *Planning Poker* e Aprendizado de Máquina)?”. Algumas das respostas são exibidas no Quadro 10.

Quadro 10 – Percepções quanto a combinação de técnicas (Acadêmicos)

Resposta
Ajuda a ter uma melhor ideia do tempo real empreendido em uma tarefa, principalmente para pessoas que ainda não possuem muita experiência na área.
Achei muito válido para ajudar o desenvolvedor com uma base de tarefas parecidas para a estimativa, é realmente muito difícil estimar uma tarefa que o mesmo não tem nenhum tipo de parâmetro ou experiência para se basear, a ferramenta ajuda bastante nesse quesito
Achei um bom par, pois muitas vezes no Planning Poker, por falta de experiência, estima-se um valor muito mais alto para uma tarefa. Em vez da correção de estimativa ser feita por um integrante da equipe com mais experiência a proposta faz esse papel.
Sem dúvida traz uma certeza maior no momento da estimativa, uma vez que você tem, ao mesmo tempo, a experiência da equipe colocada em perspectiva e o esforço médio na execução de tarefas semelhantes realizadas por outros profissionais.
Acho que é uma combinação positiva, justamente por auxiliar muito a estimativa de esforço do software trazendo informações a mais para a discussão.
Acredito que o aprendizado de máquina não virá a substituir a estimativa clássica do planning poker por imprecisões e destoancias diversas de amostragens limitadas, mas serve de grande

utilidade como ferramenta auxiliar nos debates entre equipes de planejamento.
Acho muito bacana a proposta de aprimorar o processo, respeitando o tempo necessário para cada tarefa. Facilitando para o desenvolvedor.
A combinação das técnicas consegue nos trazer resultados mais reais e fiéis, facilitando a estimação e resultando em uma excelência operacional
Acredito que a estimativa feita por machine learning é uma boa adição para a dinâmica de planning poker da equipe, dá uma outra perspectiva.

Fonte: Autoria Própria (2021)

Os comentários dos participantes direcionam para uma visão positiva a combinação de técnicas, acreditam que ela permita trazer resultados melhores, facilitando o processo de estimativas. Ao mesmo tempo defendem que o AM não virá a substituir as técnicas que valorizam o fator humano, mas se utilizadas em conjunto podem trazem resultados positivos.

5.5.2 Profissionais de TI

Para a avaliação com profissionais de TI elaboramos um questionário de busca (coletando nome e contato) por voluntários que utilizam o *Planning Poker* para as estimativas das tarefas de software. Além disso, conversamos com profissionais das empresas de TI da cidade de Chapecó, local de residência do pesquisador na procura por participantes.

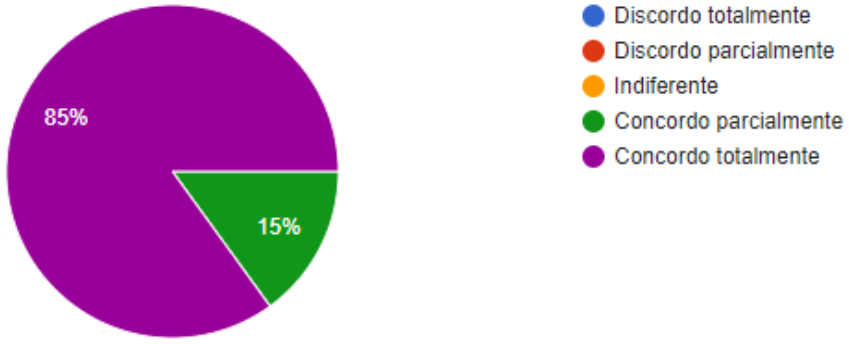
O pesquisador organizou reuniões em duas datas, permitindo maior flexibilidade para participação dos profissionais. As datas foram nos dias 29/10/2021 e 30/10/2021. Em ambas reuniões foi introduzida a ideia, explicado o fluxo da *ML Planning Poker* (Figura 45), o esboço do projeto na prática (Figura 44) e posteriormente apresentada a ferramenta desenvolvida. Foram mostradas as funcionalidades, seu processo de utilização e solicitado aos participantes a criação de uma conta e posterior utilização para experimentação. Por fim, aplicamos o questionário aos profissionais de TI, disponível no Apêndice A. Ao responder o questionário, primeiramente havia a explicação do estudo e a apresentação de um termo livre e esclarecido (Apêndice D), possibilitando aos profissionais optarem na participação ou não do estudo. A Q1 referiu-se a descobrir se o usuário respondente era de fato alguém com conta válida, visando garantir a veracidade sobre os resultados obtidos.

A Q2 referia-se ao local de residência dos profissionais de TI. Ao todo conseguimos 20 participantes; destes, 16 de Santa Catarina (15 de Chapecó, 1 de

Xanxerê), 2 do Paraná (1 de Cascavel, 1 de Francisco Beltrão), 1 de São Paulo da cidade de São José dos Campos e 1 profissional residente no exterior, em Portugal.

No que diz respeito a Q3: “Qual é sua percepção quanto a afirmação: A proposta é fácil de ser compreendida.”, os resultados aparecem no Gráfico 6.

Gráfico 6 - Resultado da Q3 (Profissionais de TI)

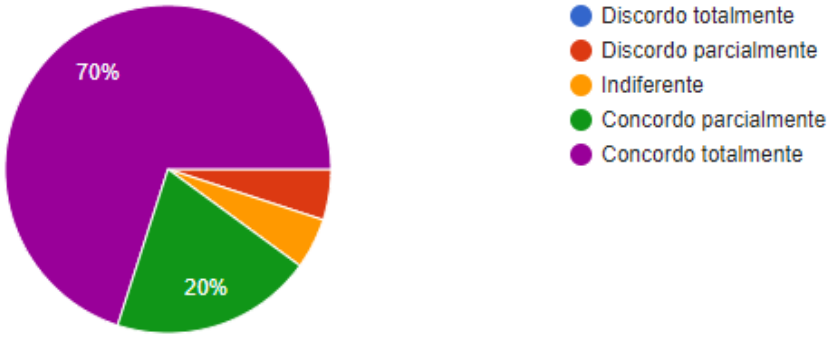


Fonte: Autoria Própria (2021)

As respostas colhidas direcionam para a facilidade de compreensão da proposta, considerando-se que todos concordam parcial ou total com a afirmação. Ter o entendimento da proposta é o primeiro passo para uma aplicação efetiva nas atividades diárias, mostrando-se positiva em termos de compreensibilidade.

No que diz respeito a Q4: “Qual é sua percepção quanto a afirmação: A proposta é fácil de ser utilizada.”, sendo apresentado no Gráfico 7.

Gráfico 7 - Resultado da Q4 (Profissionais de TI)



Fonte: Autoria Própria (2021)

As percepções apontam para facilidade de utilização da proposta com 90% dos resultados. Isso é fundamental nas empresas de TI, pois, o princípio ágil “foco nas pessoas mais do que no processo ou ferramentas” continua sendo seguido.

No que diz respeito a Q5: “Quais são as percepções positivas quanto ao uso da proposta?”. Algumas das respostas são apresentadas no Quadro 11.

Quadro 11 – Percepções positivas a proposta (Profissionais de TI)

Resposta
Todos os pontos apresentados são positivos, mas o que mais me chamou a atenção é trazer um histórico de tarefas semelhantes, isto ajuda muito principalmente quando a média da votação dos participantes tem divergência muito grande da estimada por machine learning. Nestes casos, agiliza o processo de debater as tarefas mais complexas e não perder tempo em tarefas simples.
Pode agregar muito valor, principalmente para times grandes, que tem variedade na experiência dos integrantes, como um sênior com anos de experiência, e um Junior recém contratado. A ferramenta pode dar um "norte" para todos integrantes, independente do nível de experiência.
Tendo uma máquina que leva em consideração apenas o histórico já desenvolvido, as estimativas muito "fora da curva" tendem a diminuir... Os participantes do planning podem se esquecer de uma tarefa semelhante desenvolvida tempos atrás que poderia ser utilizada como base nessa estimativa, nesses casos a ferramenta iria auxiliar bastante.
A proposta é interessante trazendo o ML como um auxilio da equipe para estimativa das tarefas. Vejo o modelo como o membro "histórico da equipe" ou sênior que vai ter guardado a estimativa de todas as tarefas feitas pela empresa.
Muitas vezes fica difícil ter um balizador na estimativa, pois geralmente temos seniores e juniores, esta proposta traz um viés neutro e se justificando com as tarefas, enriquecendo muito os debates.
A proposta é muito boa. Considerando que muitas equipes de desenvolvimento utilizam o Planning Poker, existem casos em que acontecem divergências entre decisões de tempo das tarefas, isso podendo acontecer por N fatores. Entretanto, utilizando o Aprendizado por Máquina, pode ajudar a equipe nas discussões sobre os motivos de tempo escolhido por cada integrante, levanto a um caminho mais "feliz". Isso pelo fato de ter a IA como "integrante", possuindo dados históricos de cada tarefa trazendo uma perspectiva diferente para a equipe. Esse é um trabalho perfeito para uma IA, já que nós não somos capazes de lembrar de todas as tarefas e tempo levado sobre elas.

Fonte: Autoria Própria (2021)

As percepções dos participantes direcionam para benefícios da proposta no processo de estimativas, o AM pode dar subsídio aos integrantes, independentemente do nível de experiência. Reforçaram também, a questão do esquecimento das tarefas muito antigas o que dificulta a estimativa atual e a proposta permite relembrar tarefas semelhantes desenvolvidas anteriormente. Ter a inteligência artificial como “integrante” da equipe foi visto com bons olhos permitindo o enriquecimento das discussões e resultados de estimativas mais precisos, proporcionando a melhoria do processo.

No que diz respeito a Q6: “Quais são as percepções negativas quanto ao uso da proposta?”, algumas das respostas compiladas estão no Quadro 12.

Quadro 12 – Percepções negativas a proposta (Profissionais de TI)

Resposta
Tarefas possuem particularidades que devem ser consideradas... ou seja, uma tela de Login. a IA pode estimar ABC. mas na real existem telas de Login com somente o campo, e telas que tem Validação de CPF, com subvalidações e afins.
Não seria um ponto negativo e sim uma sugestão, adicionar story points estimados e praticados no machine learning, este tipo de métrica é muito utilizada também.
O tempo para levantar um "dataset" pode ser muito alto, além da definição bem delicada dos campos estruturados que serão levados em consideração na análise.
Talvez falte parametrizar mais algumas informações para apurar melhor os resultados.
Vejo como necessária a integração com ferramentas do mercado utilizado para gestão de projetos / planejamento (redmine, jira e etc). Me desagrada ao fato de ter que lançar as informações dentro da plataforma separado das ferramentas já utilizadas dentro das equipes de desenvolvimento.
Para uma equipe de desenvolvimento nova ou que não possuí uma quantia considerável de

histórico de tarefas armazenados em algum lugar, parece ser um processo um pouco demorado para que o Aprendizado por Máquina comece a dar um resultado mais aproximado da realidade da equipe. Talvez nesses casos, a equipe iria precisar ir adaptando as opções apresentadas no projeto (de Verbo e Substantivos por exemplo).
Falta de adequação ao modelo de criação de user stories, com verbos e substantivos fixos em vez da notação comum "As a user, I want to... ". - Adequação a um modelo de tarefas padrão da proposta e não adaptável às tarefas específicas de diferentes empresa e projetos. - Modelo não aderente a sistemas populares de gerenciamento de tarefas como o JIRA. - Uso apenas na linguagem Português do Brasil.

Fonte: Autoria Própria (2021)

Percebe-se que situações direcionadas ao tempo alto para as empresas construírem seus próprios *datasets*, além de uma definição delicada no que diz respeito às *features* (variáveis descritivas para a construção do modelo de AM). Outras considerações sugerem a integração a ferramentas já disponíveis como JIRA e Redmine, permissão de cadastro de outras métricas bastante utilizadas, como o uso de *story points* e adequação do modelo para criação de *user stories*.

No que diz respeito a Q7: “Qual a sua percepção quanto a combinação de técnicas para estimativa de esforço de software (neste caso Planning Poker e Aprendizado de Máquina)?”. Algumas das respostas são exibidas no Quadro 13.

Quadro 13 – Percepções da combinação de técnicas (Profissionais de TI)

Resposta
Tenho mais percepção quanto ao poker planning mas machine learning tem um futuro muito promissor e trazer isto junto ao planejamento do desenvolvimento de softwares é muito interessante, vejo que irá ajudar muito equipes iniciais que não tiveram ou tem pouca experiência com o poker planning.
Achei legal a combinação de técnicas; O aprendizado de máquina possui uma média o que ajuda a identificar desvios padrões; A média entre as pessoas mais experientes e menos experientes é importante já que misturam valores estimados contrários em virtude do otimismo ou pessimismo com a experiência da atividade;
Acho uma combinação muito interessante pois o ML pode trazer uma maior assertividade na votação e com isso a equipe tende a erra menos nas estimativas e ter um controle maior sobre o tempo das tarefas.
Sem dúvidas é muito importante, pois as equipes tendem a ter pouca experiência, devido rotatividade nas empresas, com isso o software vem agregar. Outro ponto importante é que ao aplicar este modelo a empresa vai prestar muita atenção nas métricas e processos, tendendo a evoluir cada vez mais o processo de estimativa.
Todas as técnicas que possam auxiliar são bem vindas, pois trata-se de um assunto sensível, a estimativa, que varia muito de time pra time, empresa para empresa, negócio para negócio. Então sempre que algo possa a agregar e colaborar para ser mais assertivo nas estimativas isso merece atenção e deve ser experimentado.
Acho muito interessante, pois atualmente a inteligência artificial está presente em muitas ferramentas e auxilia de forma significativa as pessoas. A questão de mesclar engenharia de software com aprendizado de máquina é uma excelente estratégia para desenvolvimento das sprints dos times.
A ideia é muito bacana tanto para pequenas, quanto grandes equipes. Com o tempo, o aprendizado, os registros de horas, históricos de tarefas, podem ser muito úteis durante as discussões e decisões em um planning. O Planning Poker já está no dia a dia, porém a técnica de estimar o esforço não está, contudo, com certeza vem para agregar.

Fonte: Autoria Própria (2021)

Os comentários dos participantes direcionam para uma visão positiva a combinação de técnicas, defendendo que tudo o que vem a agregar nas estimativas e reduz dúvidas deve ser utilizado em favor do processo.

Neste capítulo descrevemos todas as etapas da implementação, documentação e avaliação da proposta, bem como os detalhes de cada fase. E por fim, em linhas gerais, as percepções coletadas em torno da *ML Planning Poker* apontam para a possibilidade de utilização em cenários reais, conta com aspectos positivos no que diz respeito à ideia e adesão por partes de equipes devido a facilidade de utilização e potencial presenciado e relatado quanto à melhoria o processo de estimativas. Ao mesmo tempo, existem possibilidades de aperfeiçoamento, sugestões para um maior amadurecimento e evolução da abordagem.

6 CONSIDERAÇÕES FINAIS

O presente trabalho apresentou uma problemática existente no processo de estimativas de esforço de software, mais especificamente, o objetivo desta pesquisa foi descobrir se combinar *Planning Poker* e aprendizado de máquina interfere no processo de estimativas. A aplicação da proposta criada e nomeada *ML Planning Poker* nos permitiu a coleta de percepções de estudantes de graduação e profissionais de TI avaliando os principais aspectos da abordagem.

Os resultados apontaram para a assertividade (igualdade no tempo estimado *versus* realizado) em 24,3% em ambas as abordagens. Porém, no que se refere as tarefas com estimativa incorreta, percebemos que a proposta teve melhor resultado, já que, 57,1% das tarefas teve diferença de no máximo 1 hora entre tempo estimado e realizado em comparação a 39,2% do *Planning Poker* original. Deste modo também, analisamos que 60,7% das tarefas do *Planning Poker* original teve erro maior de 1 hora em comparação a 42,8% da proposta.

Os questionários com os estudantes apontam que 81,2% concordam que a ferramenta contribui com o processo de estimativas, isso nos permite considerar que a combinação de *Planning Poker* e aprendizado de máquina trouxe uma percepção positiva quanto ao uso.

Mesmo que hajam questões a serem melhoradas em torno da proposta, percepções direcionam para pontos positivos nas tarefas em que os membros não possuem experiência, haja vista que a proposta traz atividades similares realizadas, trazendo segurança maior ao participante na definição de sua estimativa. Além disso, estes exemplos similares podem reduzir o tempo de debate auxiliando para atingir um consenso na estimativa.

Os profissionais de TI levantaram benefícios junto ao processo de estimativas, o AM pode dar um subsídio aos integrantes, independentemente do nível de experiência. Complementaram também a questão do esquecimento das tarefas muito antigas, o que dificulta a estimativa atual e a proposta permite relembrar tarefas semelhantes desenvolvidas anteriormente.

Limitações nos experimentos ocorreram devido à pandemia de Covid que não permitiu um processo de avaliação presencial. Além disso, o fato de avaliar com estudantes de graduação que estão em processo de aprendizado, e usando um

dataset (conjunto de dados) com estimativas de outros profissionais pode não ter permitido coletar os resultados mais precisos sobre a proposta. Uma dificuldade encontrada é a baixa quantidade de *datasets* públicos disponíveis, havendo assim, a necessidade da disponibilização de outros conjuntos de dados que facilitem a realização de novos estudos referentes as estimativas de esforço de software.

Como trabalhos futuros podem ser construídos outros modelos de AM sobre *datasets* de empresas para que posteriormente testes sejam realizados sobre estes dados, representando assim a realidade da capacidade de estimar por parte dos membros da equipe que estão realizando o processo.

A combinação proposta neste trabalho mostra-se com potencial para a aplicação em cenários reais permitindo contribuir com o processo de estimativas, este tema ainda investigável. A crescente do uso do AM em diversos âmbitos nos permite acreditar que a tendência seja de utilizá-lo no desenvolvimento de software e no processo de estimativas não deverá ser diferente, especialmente no apoio à tomada de decisões.

REFERÊNCIAS

ABDELLATIF, Tamer Mohamed. A comparison study between soft computing and statistical regression techniques for software effort estimation. In: **IEEE.2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)**. [S.l.], 2018. p. 1–5.

ALPAYDIN, Ethem. **Introduction to machine learning.** [S.l.]: MIT press, 2020.

BARDSIRI, V Khatibi; JAWAWI, Dayang Norhayati Abang; HASHIM, S Zaiton Mohd; KHATIBI, Elham. Increasing the accuracy of software development effort estimation using projects clustering. **IET software**, IET, v. 6, n. 6, p. 461–473, 2012.

BASILI, Victor R; LINDVALL, Mikael; COSTA, Patricia. Implementing the experience factory concepts as a set of experience bases. In: **SEKE.** [S.l.: s.n.], 2001. p. 102–109.

BENALA, Tirimula Rao; DEHURI, Satchidananda; MALL, Rajib. Computational intelligence in software cost estimation: an emerging paradigm. **ACM SIGSOFT Software Engineering Notes**, ACM New York, NY, USA, v. 37, n. 3, p. 1–7, 2012.

BLOCH, Michael; BLUMBERG, Sven; LAARTZ, Jürgen. Delivering large-scale it projects on time, on budget, and on value. **Harvard Business Review**, p. 2–7, 2012.

BØRTE, Kristin; LUDVIGSEN, Sten R; MØRCH, Anders I. The role of social interaction in software effort estimation: Unpacking the “magic step” between reasoning and decision-making. **Information and Software Technology**, Elsevier, v. 54, n. 9, p. 985–996, 2012.

BRASILEIRO, Roberto. **Planning Poker: A melhor maneira de estimar qualquer atividade**. 2017. Método Ágil. Disponível em: <https://www.metodoagil.com/planning-poker>

CHAPMAN, Pete; CLINTON, Julian; KERBER, Randy; KHABAZA, Thomas; REINARTZ, Thomas; SHEARER, Colin; WIRTH, Rudigeret al. Crisp-dm 1.0: Step-by-step data miningguide. **SPSS inc**, v. 9, p. 13, 2000.

CHEN, Tianqi; GUESTRIN, Carlos. Xgboost: A scalable tree boosting system. In: **Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining.** 2016. p. 785-794.

COHN, Mike. **Agile estimating and planning.** [S.l.]: Pearson Education, 2005.

DANTAS, Emanuel; PERKUSICH, Mirko; DILORENZO, Ednaldo; SANTOS, Danilo FS; ALMEIDA, Hyggo; PERKUSICH, Angelo. Effort estimation in agile software development: An updated review. **International Journal of Software Engineering and Knowledge Engineering**, World Scientific, v. 28, n. 11n12, p. 1811–1831, 2018.

ELISH, Mahmoud O. Assessment of voting ensemble for estimating software development effort. In: IEEE.**2013 IEEE Symposium on Computational Intelligence and Data Mining(CIDM).** [S.l.], 2013. p. 316–321.

EMAM, Khaled El; KORU, A Günes. A replicated survey of it software project failures. **IEEE software**, IEEE, v. 25, n. 5, p. 84–90, 2008.

FACELI, Katti; LORENA, Ana Carolina; GAMA, João; CARVALHO, André Carlos Ponceде Leon et al. Inteligência artificial: Uma abordagem de aprendizado de máquina. **Rio de Janeiro: LTC**, 2011.

GERHARDT, Tatiana Engel; SILVEIRA, Denise Tolfo. **Métodos de pesquisa.** [S.l.]: Plageder,2009.

GONZÁLEZ-CARRASCO, Israel; COLOMO-PALACIOS, Ricardo; LÓPEZ-CUADRADO, José Luis; PEÑALVO, Francisco José García. Seffest: Effort estimation in software projects using fuzzy logic and neural networks. **International Journal of Computational Intelligence Systems**, Taylor & Francis, v. 5, n. 4, p. 679–699, 2012.

GRENNING, James. Planning poker or how to avoid analysis paralysis while release planning. **Hawthorn Woods: Renaissance Software Consulting**, v. 3, p. 22–23, 2002.

HICKS, Seann. **Planning Poker - An Illustred Guide.** 2020. Blog. Disponível em: <https://www.seannhicks.com/blog/planning-poker/>.

HOSNI, Mohamed; IDRI, Ali; NASSIF, Ali Bou; ABRAN, Alain. Heterogeneous ensembles for software development effort estimation. In: IEEE. **2016 3rd International Conference on Soft Computing & Machine Intelligence (ISCMI).** [S.l.], 2016. p. 174–178.

IDRI, Ali; HOSNI, Mohamed; ABRAN, Alain. Systematic literature review of ensemble effort estimation. **Journal of Systems and Software**, Elsevier, v. 118, p. 151–175, 2016.

JØRGENSEN, Magne. A review of studies on expert estimation of software development effort. **Journal of Systems and Software**, Elsevier, v. 70, n. 1-2, p. 37–60, 2004.

KELLEHER, John D; NAMEE, Brian Mac; D'ARCY, Aoife. **Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies.** [S.l.]: MIT press, 2015.

KOCAGUNELI, Ekrem; MENZIES, Tim; KEUNG, Jacky W. On the value of ensemble effort estimation. **IEEE Transactions on Software Engineering**, IEEE, v. 38, n. 6, p. 1403–1416,2011.

LAKATOS, Eva Maria; MARCONI, M de A. Fundamentos de metodologia científica. 5. reimp. **São Paulo: Atlas**, v. 310, 2007.

LIMA, Telma Cristiane Sasso de; MIOTO, Regina Célia Tamaso. Procedimentos metodológicos na construção do conhecimento científico: a pesquisa bibliográfica. **Revista Katálysis**, SciELOBrasil, v. 10, n. SPE, p. 37–45, 2007.

MAHNIC, Viljan; HOVELJA, Tomaz. On using planning poker for estimating user stories. **Journal of Systems and Software**, Elsevier, v. 85, n. 9, p. 2086–2095, 2012.

MANIFESTO ÁGIL. **Manifesto para Desenvolvimento Ágil de Software.** 2001. Disponível em:< https://agilemanifesto.org/iso/ptbr/manifesto.html>. Acessado em: 08/10/2021.

MEYER, Bertrand. The software knowledge base. In: **ICSE.** [S.l.: s.n.], 1985. p. 158–165.

MOHARRERI, Kayhan; SAPRE, Alhad Vinayak; RAMANATHAN, Jayashree; RAMNATH, Rajiv. Cost-effective supervised learning models for software effort estimation in agile environments. In: **IEEE. 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC).** [S.l.], 2016. v. 2, p. 135–140.

MORDE, Vishal. **Xgboost algorithm: Long may she reign.** Towards Data Science, 2019.

NASSIF, Ali Bou; HO, Danny; CAPRETZ, Luiz Fernando. Towards an early software estimation using log-linear regression and a multilayer perceptron model. **Journal of Systems and Software**, Elsevier, v. 86, n. 1, p. 144–160, 2013.

NGUYEN-CONG, Danh; TRAN-CAO, De. A review of effort estimation studies in agile, iterative and incremental software development. In: IEEE. **The 2013 RIVF International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for Future (RIVF)**. [S.l.], 2013. p. 27–30.

PARK, Heejun; BAEK, Seung. An empirical validation of a neural network model for software effort estimation. **Expert Systems with Applications**, Elsevier, v. 35, n. 3, p. 929–937, 2008.

PMBOK, GUIA. Um guia do conjunto de conhecimentos em gerenciamento de projetos. Sexta edição. **Project Management Institute, Inc. EUA**, 2017.

POSPIESZNY, Przemysław; CZARNACKA-CHROBOT, Beata; KOBYLI´NSKI, Andrzej. Application of function points and data mining techniques for software estimation-a combined approach. In: **Software Measurement.** [S.l.]: Springer, 2015. p. 96–113.

PRESSMAN, Roger S. **Engenharia de Software: Uma abordagem profissional. Setima Edição. [S.l.]**: São Paulo: Pearson Makron Books, 2011.

PROVOST, Foster; FAWCETT, Tom. Data science and its relationship to big data and data-driven decision making. **Big data**, Mary Ann Liebert, Inc. 140 Huguenot Street, 3rd Floor New Rochelle, NY 10801 USA, v. 1, n. 1, p. 51–59, 2013.

QI, Kan; HIRA, Anandi; VENSON, Elaine; BOEHM, Barry W. Calibrating use case points using bayesian analysis. In: **Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement.** [S.l.: s.n.], 2018. p. 1–10.

RAMOS, Vinicius de A. **Desenvolvimento Web com Python e Django: Introdução.** 2018. Python Academy. Disponível em: <https://pythonacademy.com.br/blog/desenvolvimento-web-com-python-e-django-introducao>.

RASTOGI, Himani; DHANKHAR, Swati; KAKKAR, Misha. A survey on software effort estimation techniques. In: IEEE. **2014 5th International Conference-Confluence The Next Generation Information Technology Summit (Confluence).** [S.l.], 2014. p. 826–830.

RUS, Ioana; LINDVALL, Mikael; SINHA, S. Knowledge management in software engineering. **IEEE software**, v. 19, n. 3, p. 26–38, 2002.

RUSSEL, Stuart; NORVIG, Peter. **Inteligência Artificial.** [S.l.]: Tradução da Terceira Edição. Elsevier Editora, 2013.

SARAÇ, Ömer Faruk; DURU, Nevcihan. A novel method for software effort estimation: Estimating with boundaries. In: IEEE. **2013 IEEE INISTA.** [S.l.], 2013. p. 1–5.

SCIKIT-YB. **Classification Report.** Yellowbrick (Online). Disponível em:< https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html>. Acessado em: 16/12/2021.

SHALEV-SHWARTZ, Shai; BEN-DAVID, Shai. **Understanding machine learning: From theory to algorithms.** [S.l.]: Cambridge university press, 2014.

SHARMA, Pinkashia; SINGH, Jaiteg. Systematic literature review on software effort estimation using machine learning approaches. In: **IEEE.2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS).** [S.l.], 2017. p. 43–47.

TISSOT, Andre Augusto; EMER, Maria Cláudia Figueiredo Pereira; BASTOS, Laudelino Cordeiro. Influence of the review of executed activities utilizing planning poker. In: IEEE. **2015 29th Brazilian Symposium on Software Engineering.** [S.l.], 2015. p. 170–178.

TRONTO, Iris Fabiana de Barcelos; SILVA, José Demísio Simões da; SANT'ANNA, Nilson. An investigation of artificial neural networks based prediction systems in

software project management. **Journal of Systems and Software**, Elsevier, v. 81, n. 3, p. 356–367, 2008.

USMAN, Muhammad; MENDES, Emilia; BÖRSTLER, Jürgen. Effort estimation in agile software development: a survey on the state of the practice. In: **Proceedings of the 19thinternational conference on Evaluation and Assessment in Software Engineering.** [S.l.:s.n.], 2015. p. 1–10.

WEN, Jianfeng; LI, Shixian; LIN, Zhiyong; HU, Yong; HUANG, Changqin. Systematic literature review of machine learning based software development effort estimation models. **Information and Software Technology**, Elsevier, v. 54, n. 1, p. 41–59, 2012.

ZIAUDDIN, Shahid Kamal Tipu; ZIA, Shahrukh. An effort estimation model for agile software development. **Advances in computer science and its applications (ACSA)**, v. 2, n. 1, p.314–324, 2012.

APÊNDICE A - Questionários de Avaliação

Formulário de Avaliação aos Estudantes Participantes

1) - Qual é seu usuário de acesso à ferramenta? (OBS: Esta resposta não será utilizada nos resultados. Desejamos nos precaver de que apenas usuários que de fato criaram conta respondam a pesquisa).

2) - Qual é sua percepção quanto a afirmação: O debate ocorrido no Planning Poker é prejudicado com o uso da proposta.
( ) discordo totalmente ( ) discordo parcialmente ( ) indiferente ( )concordo parcialmente ( )concordo totalmente

3) - Qual é sua percepção quanto a afirmação: O Planning Poker perde suas características essenciais com o uso da proposta.
( ) discordo totalmente ( ) discordo parcialmente ( ) indiferente ( )concordo parcialmente ( )concordo totalmente

4) - Qual é sua percepção quanto a afirmação: A combinação do Planning Poker com a proposta contribui no processo de estimativas.
( ) discordo totalmente ( ) discordo parcialmente ( ) indiferente ( )concordo parcialmente ( )concordo totalmente

5) - Qual é sua percepção quanto a afirmação: A proposta é fácil de ser compreendida.
( ) discordo totalmente ( ) discordo parcialmente ( ) indiferente ( )concordo parcialmente ( )concordo totalmente

6) - Qual é sua percepção quanto a afirmação: A proposta é fácil de ser utilizada.
( ) discordo totalmente ( ) discordo parcialmente ( ) indiferente ( )concordo parcialmente ( )concordo totalmente

7) - Quais são as percepções positivas quanto ao uso da proposta?
R:

8) - Quais são as percepções negativas quanto ao uso da proposta?
R:

9) - Qual a sua percepção quanto a combinação de técnicas para estimativa de esforço de software (neste caso Planning Poker e Aprendizado de Máquina)?
R:

Formulário de Avaliação aos Profissionais de TI

1) - Qual é seu usuário de acesso à ferramenta? (OBS: Esta resposta não será utilizada nos resultados. Desejamos nos precaver de que apenas usuários que de fato criaram conta respondam a pesquisa).

2) - Qual é sua percepção quanto a afirmação: A proposta é fácil de ser compreendida.
( ) discordo totalmente ( ) discordo parcialmente ( ) indiferente ( )concordo parcialmente ( )concordo totalmente

3) - Qual é sua percepção quanto a afirmação: A proposta é fácil de ser utilizada.
( ) discordo totalmente ( ) discordo parcialmente ( ) indiferente ( )concordo parcialmente ( )concordo totalmente

4) - Quais são percepções positivas quanto ao uso da proposta?
R:

5) - Quais são percepções negativas quanto ao uso da proposta?
R:

6) - Qual a sua percepção quanto a combinação de técnicas para estimativa de esforço de software (neste caso Planning Poker e Aprendizado de Máquina)?
R:

APÊNDICE B - *Features* usadas no modelo de ML

Quadro 14 – Variáveis descritivas do modelo

Verbo inicial		**Categoria**
1 – Adicionar	18 – Organizar	1 – Desenvolvimento: 621
2 – Aprender	19 – Permitir	2 – Gestão : 138
3 – Atualizar	20 – Pesquisa	3 – Operacional: 283
4 – Automatizar	21 – Remover	
5 – Backup	22 – Reunião	**Subcategoria**
6 – Configurar	23 – Semana	1 – Aprimoramento: 271
7 – Converter	24– Sincronizar	2 – Consultoria: 15
8 – Corrigir	25 – Suporte	3 – Documentação: 18
9 – Criar	26 – Testar	4 – Erro: 138
10 – Definir	27 – Treinamento	5 – Lançamento: 45
11 – Escrever	28 – Validar	6 – Marketing: 61
12 – Instalar	29 – Verificar	7 – Reunião de Gestão: 46
13 – Investigar		8 – Reunião de progresso: 23
14 – Melhorar		9 – Suporte ao Cliente: 101
15 – Mesclar		10 – Suporte Interno: 253
16 – Migrar		11 – Teste: 34
17 – Obter		12 – Treinamento: 37
SUBSTANTIVOS		
0 – Acesso		
1 – Adição	2 – Alinhamento	3 – Alterações
4 - Ambiente	5 - Ambiente de Desenvolvimento	6 - Ambiente de Hospedagem
7 – Arquivos	8 - Arrastar	9 – Assinaturas
10 – Atualização	11 - Atualização Automática	12 – Auditoria
13 – Backup	14 – Banco	15 – Botão
16 – Busca	17 – C#	18 - C# e SQL
19 – CSV	20 – Caixa	21 - Caixa de Seleção
22 – Caixa de Texto	23 – Calendário	24 – Campo
25 – Campos	26 – Carregador	27 - Casas decimais
28 – Certificado	29 – Chave	30 – Classes
31 – Coluna	32 – Comparador	33 – Conexão
34 – Configuração	35 - Consulta	36 – Conta
37 – Controle	38 - Controle de Guia	39 - Controle de Notas
40 - Controle de Pesquisa	41 – Conversão	42 – Correções
43 - Criação	44 – Critérios de Pesquisa	45 – Cronograma
46 – Custos	47 – Cálculo	48 – Código
49 – Dados	50 – Data	51 – Demonstração

52 – Descrição	53 – Desempenho	54 – Desenvolvedor
55 – Design	56 – Discrepâncias	57 – Divisores
58 – Documentação	59 – Download	60 – Drivers
61 – Email	62 – Entrada	63 – Equipe
64 – Erro	65 – Erros	66 – Escalas
67 – Escritórios	68 – Estatística	69 – Estrutura
70 – Exceção	71 – Exclusão	72 – Exportação
73 – Extensão	74 – Figuras	75 – Filtro
76 – Firewall	77 – Fluxo	78 – Formatação
79 - Formulário	80 – Formulários	81 – Foto
82 - Front-end	83 – Funcionalidade	84 – Funcionalidade de Email
85 – Gerador	86 – Gestão	87 – Grade
88 – Guia	89 – Histórico	90 – Horários
91 – Imagens	92 – Implantação	93 – Importador
94 - Importador de Arquivos	95 – Impressão	96 – Instalação
97 – Integridade	98 – Itens	99 – LAN
100 – Lançamento	101 – Laptop	102 – Licença
103 – Limites	104 – Lista	105 – Livro de Figuras
106 – Login	107 – Logo	108 – Manual
109 – Marketing	110 – Material	111 – Mensagem
112 – Menu	113 - Menu e Barra de Ferramentas	114 – Migração
115 – Moedas	116 – Movimento	117 – Mudanças
118 – Nome	119 - Nova Pesquisa	120 – Números
121 - Objeto de Negócio	122 – Office	123 – Opção
124 – Ordem	125 – Pagamento	126 – Painéis
127 – Particular	128 – Pendências	129 – Permissões
130 – Pesquisa	131 - Pesquisa Adicional	132 - Pesquisa Automática
133 – Planilha	134 – Plano	135 - Plug-in
136 – Política	137 – Procedimentos	138 – Processamento
139 – Programa	140 – Progresso	141 – Propostas
142 – Página	143 – Recurso	144 – Rede
145 – Referência	146 – Referências	147 – Regressão
148 – Relatório	149 – Relatórios	150 - Renovação de Licenças
151 – Repositório	152 – Requisitos	153 – Resolução
154 – Reunião	155 – Revisão	156 – Robustez
157 – Rota	158 – Roteamento	159 – SQL
160 - SQL Server	161 – Salvamento	162 –Saúde

163 - Script	164 – Segurança	165 – Semana
166 – Servidor	167 – Setup	168 – Seção
169 – Sistema	170 - Sistema (Total)	171 – Tabela
172 – Tabelas	173 – Tarefas	174 – Tela
175 - Tempo Limite	176 – Teste	177 - Teste de Estresse
178 - Teste de Liberação	179 – Testes	180 – Texto
181 - Tipos de Documentos	182 - Tratamento de Erros	183 – Travamento
184 – Twitter	185 – Técnica	186 -Técnica Estendida
187 – Título	188 – Usuário	189 – VPN
190 –Validação	191 - Vazamento de Memória	192 – Velocidade
193 – Versão	194 - Visual Studio	195 - Vários Documentos
196 – Website	197 – Windows	

Fonte: Autoria Própria (2021)

APÊNDICE C - Tarefas desenvolvidas na avaliação antes/depois

Tabela 6 - Processo de Estimativas usando o *Planning Poker* original

ESTIMATIVAS USANDO *PLANNING POKER* ORIGINAL

ID	Tarefa	Tempo Estimado	Tempo Realizado
1	Adicionar funcionalidade de cadastramento	5	6
2	Modelar diagramas de casos de uso	4	3
3	Criar filtro para a exibição dos monitores	1	5
4	Implementar interface para login e cadastramento	3	5
5	Escrever relatório do sistema	7	10
6	Função de buscar, adicionar, atualizar receitas no DB (GET, POST e PATCH)	5	1
7	Função de criar, buscar e atualizar cardápio no DB (POST, GET e PATCH)	3	3
8	Função de buscar, criar e atualizar informações dos funcionários (GET, POST e PATCH)	3	1
9	Função para criar, buscar e atualizar restaurantes (POST, GET e PATCH)	3	1
10	Função de adicionar, buscar e atualizar o status de um pedido (POST, GET e PATCH)	3	4
11	Adicionar conexão com o banco	0,5	0,5
12	Definição da identidade visual do sistema	3	1
13	Criação da tela de cadastro de novos clientes	5	2,5
14	Criação de tela de login	3	2,25
15	Testes unitários para validação das funções	5	1,5
16	Diagramas de casos de uso	5	3
17	Criar da estrutura do banco de dados	2	0,5
18	Estruturar diagrama de classes UML	5,5	2,5
19	Realizar testes unitários	4	3
20	Criar diagrama de caso de uso	4	2
21	Criar interface gráfica de interação	2	4
22	Criar funcionalidade de cadastro de usuário	2	1
23	Configuração do sistema	2	2
24	Criação do login p/ funcionário e gerente	3	2
25	Criação das telas principais	2	2
26	Criação da inserção/remoção/edição de funcionários	3	3
27	Criação da inserção/remoção/edição de serviços	3	3
28	Elaboração do diagrama UML, diagrama de casos de uso e diagrama de sequência	9	11
29	Criação da tela de login e usuário	2	2
30	Criação da interface gráfica principal	6	5
31	Criação das tabelas, relações do banco de dados e configuração de conexão com o banco	7	7
32	Implementação da funcionalidade de login/usuário	5	4

	com banco de dados		
33	Povoamento do banco de dados	5	4,5
34	Criação do serviço de cadastro de usuários	4	4
35	Criação de tabelas do banco de dados	5	3
36	Criação de funcionalidade de impressão de ticket	5	4
37	Criação de tela de menu de usuário	6	3

Fonte: Autoria Própria.

Tabela 7 – Processo de Estimativas usando a *ML Planning Poker*

ESTIMATIVA USANDO A *ML PLANNING POKER*

ID	Tarefa	Tempo Estimado	Tempo Realizado
1	Instalar bibliotecas Qt e MySQL	5	5
2	Criar script do banco de dados	3	2
3	Implementar interface para exibir lista de monitores	3	6
4	Testar as funcionalidades do programa	3	6
5	Modelar Diagramas de Classes	1	1
6	Criação de tela de receitas	5	1
7	Criação de tela de cardápios	5	6
8	Criação de tela de funcionários	8	1
9	Criação de tela de restaurantes	3	5
10	Criação de tela de pedidos	5	7
11	Criação de condição para não aceitar cadastro de funcionário com mais de 8 horas de trabalho	0,5	0,5
12	Relação do sistema com o banco de dados	7	1
13	Diagrama de classes UML	2	1,5
14	Tela de contratação de serviço de lava-rápido	1,66	2
15	Tela com o monitoramento de câmeras das vagas ocupadas	6	3
16	Relatório/Documentação do sistema	21	17
17	Criar funcionalidade de login de usuário	2	5
18	Criar funcionalidade de cadastro de tarefas	2	3
19	Criar estrutura de organização por agrupamento de tarefas	1	1
20	Criar funcionalidade de criar categorias de tarefas	1,5	1
21	Criar funcionalidade de prioridade de tarefa	1,5	2
22	Criação da inserção/remoção/edição de produtos	3	3
23	Criação da tela principal (home) com as informações principais do sistema	2	2
24	Criação das models	1	1
25	Criação dos Diagramas de classes	3	3
26	Testes e experiência do usuário	2	3

27	Criação das classes (objetos modelos)	2	2
28	Implementação das funcionalidades de inserção, busca e remoção dos objetos (script SQL)	3	2,5
29	Implementação das funcionalidades de inserção, busca e remoção dos objetos de forma visível na interface	4	3
30	Implementação de tratamentos de erros para experiência do usuário	3	2
31	Elaboração do relatório	18	20
32	Revisão do código e teste massivos	4	3
33	Criação de funcionalidade de login	4	3
34	Criação de registro de veículo na garagem	3	4
35	listagem de veículos	4	3
36	Validar acesso de usuários	6	4
37	Criação de tela menu de admin	3	4

Fonte: Autoria Própria.

APÊNDICE D - Termo de Consentimento Livre e Esclarecido

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO (TCLE)

APRESENTAÇÃO DA PESQUISA: Estimar esforço de software é um fator crítico para as organizações, pois subestimativa ou superestimativa podem resultar em falhas nos projetos. No contexto da opinião especializada o Planning Poker é uma das técnicas mais utilizadas por equipes ágeis, tal que a estimativa ocorre com base na experiência dos integrantes, mediante reunião envolvendo todos os membros da equipe. Porém, as informações geradas pelo debate no Planning Poker não são guardadas devido a informalidade aparente do método e como esse conhecimento se perde, não há como aproveitá-lo em estimativas futuras. A aplicação de técnicas de aprendizado de máquina junto às estimativas de esforço cresceu nos últimos anos, usadas complementarmente ou alternativamente a outras técnicas. Alguns estudos apontam que a utilização de técnicas combinadas proporciona maior assertividade em relação a técnicas individuais.

OBJETIVO: este estudo objetiva avaliar se a combinação da técnica Planning Poker com aprendizado de máquina interfere no processo de estimativas de esforço de software.

SIGILO E PRIVACIDADE: Garantimos que sua identidade será mantida em sigilo e nenhuma informação será dada a outras pessoas. E, na divulgação dos resultados desse estudo, dados pessoais não são usados. Em caso de dúvidas sobre a pesquisa, você poderá entrar em contato com o pesquisador Doglas André Finco, através do telefone (49) 9 9124-6294 e e-mail doglas.andref@gmail.com.

Eu declaro ter conhecimento das informações contidas neste documento e ter recebido respostas claras às minhas questões a propósito da minha participação direta (ou indireta) na pesquisa e, adicionalmente, declaro ter compreendido o objetivo e benefícios deste estudo. Você livre e voluntariamente concorda em participar deste estudo?

- ◯ Sim
- ◯ Não