# MODELO PARA PRIORIZAÇÃO DO BACKLOG KANBAN DE UMA TECH BANKING

# MODEL FOR PRIORITIZING THE KANBAN BACKLOG OF A TECH BANKING

# MODELO PARA PRIORIZAR EL TRABAJO PENDIENTE DE KANBAN DE UNA TECH BANKING

Cited as:
Lima, Thiago B. A. de & Costa, Renato E. (2024). Modelo para priorização do backlog kanban de uma tech banking. Revista Gestão & Tecnologia. Journal of Management and Technology. v 24, nº 1. p.309-336

Thiago Bolivar Albuquerque de Lima
Especialista em Governança de TI e MBA em Gestão Estratégica de Inovação Tecnológica e Propriedade Intelectual
https://orcid.org/0009-0008-0227-452X

Renato Eliseu Costa
Doutor em Políticas Públicas pela Universidade Federal do ABC
https://orcid.org/0000-0001-6722-3653

Editor Científico: José Edson Lara
Organização Comitê Científico
Double Blind Review pelo SEER/OJS
Recebido em 03/09/2023
Aprovado em 08/03/2024

**RESUMO**

**Objetivo do estudo:** Este trabalho teve como objetivo desenvolver um modelo de aprendizado de máquina que faz a priorização e a sugestão de itens de "backlog Kanban", executáveis dentro de um ciclo de desenvolvimento, para um time ágil responsável pelo produto de conta corrente de um tech banking.

**Metodologia:** A pesquisa utilizou o método exploratório de natureza aplicada, seguindo a metodologia "Team Data Science Process" (Processo de ciência de dados de equipe) [TDSP] desenvolvida pela Microsoft.

**Relevância:** Com times reduzidos, causados pelo apagão de mão de obra especializada no Brasil e uma lista de demandas em espera cada vez maior, as empresas de tecnologia precisam saber o que priorizar, ter a noção do que o time é capaz de executar e ao mesmo tempo preservar a qualidade de vida dos seus colaboradores.

**Principais resultados:** A implementação com o algoritmo XGBoost se mostrou o melhor modelo supervisionado de aprendizado de máquina para realizar a estimativa dos prazos das atividades. O algoritmo final fez a sugestão das demandas na qual o time tem a capacidade de executar em um ciclo.

**Contribuições teóricas/metodológicas:** A pesquisa aponta uma deficiência nos processos de desenvolvimento de times ágeis e sugere a aplicação de inteligência artificial para cobrir essa lacuna.

**Contribuições para a gestão:** O método algoritmo proposto apoia a gestão na tomada de decisão e na previsibilidade do que o time é capaz de entregar em um período definido.

**Palavras-chave:** Priorização de backlog, AHP Gaussiano, GLMNET, XGBoost, Processamento de Linguagem Natural

**ABSTRACT**

**Objective of the study:** This work aimed to develop a machine learning model that prioritizes and suggests kanban backlog items, executable within a development cycle, for an agile team responsible for the current account product of a tech banking.

**Methodology:** The research used an exploratory method of an applied nature, following the "Team Data Science Process" [TDSP] methodology developed by Microsoft.

**Relevance:** With reduced teams, caused by the shortage of specialized labor in Brazil and an ever-increasing list of demands on hold, technology companies need to know what to prioritize, have a sense of what the team can execute and at the same time preserve the quality of life of its employees.

**Main results:** The implementation with the XGBoost algorithm proved to be the best supervised machine learning model to predict activity deadlines. The final algorithm suggested the demands that the team has the capacity to execute in a cycle, pointing out pairs of developers and testers, considering a predefined maximum deadline.

**Theoretical/methodological contributions:** The research points to a deficiency in agile team development processes and suggests the application of artificial intelligence to fill this gap.

**Contributions to management:** The proposed algorithm method supports management in decision-making and in the predictability of what the team can deliver in a defined period.

**Keywords:** Backlog prioritization, Gaussian AHP, GLMNET, XGBoost, Natural Language Processing

**RESUMEN**
**Propósito del estudio:** Este trabajo tuvo como objetivo desarrollar un modelo de aprendizaje automático que priorice y sugiera elementos del backlog kanban, ejecutables dentro de un ciclo de desarrollo, para un equipo ágil responsable del producto de cuenta corriente de una banca tecnológica.
**Metodología:** La investigación utilizó un método exploratorio de carácter aplicado, siguiendo la metodología "Team Data Science Process" [TDSP] desarrollada por Microsoft.
**Relevancia:** Con equipos reducidos, causada por la escasez de mano de obra especializada en Brasil y una lista cada vez mayor de demandas en espera, las empresas de tecnología necesitan saber qué priorizar, tener una idea de lo que el equipo es capaz de ejecutar y en al mismo tiempo preservando la calidad de vida de sus empleados.
**Principales resultados:** La implementación con el algoritmo XGBoost demostró ser el mejor modelo de aprendizaje automático supervisado para predecir los plazos de las actividades. El algoritmo final sugirió las demandas que el equipo tiene capacidad de ejecutar en un ciclo, señalando parejas de desarrolladores y probadores, considerando un plazo máximo predefinido.
**Aportes teóricos/metodológicos:** La investigación apunta a una deficiencia en los procesos ágiles de desarrollo de equipos y sugiere la aplicación de inteligencia artificial para llenar este vacío.
**Contribuciones a la gestión:** El método algorítmico propuesto apoya a la gestión en la toma de decisiones y en la previsibilidad de lo que el equipo es capaz de entregar en un período definido.

**Palabras clave:** Priorización de trabajos pendientes, AHP gaussiano, GLMNET, XGBoost, procesamiento del lenguaje natural.

## 1. INTRODUÇÃO

Grandes e médias empresas já assumem que a Tecnologia da Informação [TI] é pervasiva ao negócio. Segundo dados divulgados pela FGVcia, entre os anos de 2021 e 2022, as empresas têm investido em média 8,7% da receita em TI e esse investimento está em tendência de alta. Além disso houve uma antecipação de 1 a 4 anos do processo de transformação digital após a pandemia (Meirelles, 2022). Esse cenário reforça ainda mais a competitividade e necessidade de mudanças recorrentes nas estratégias e operações. A maioria das mudanças impactam os softwares que suportam o negócio, seja por necessidades funcionais ou por necessidades não-funcionais como segurança, escalabilidade, resiliência e outros. A natureza variável do desenvolvimento de software pode trazer lucratividade para as empresas

fornecedoras de tecnologia, por gerar um volume alto na demanda, mas também pode trazer muitos desafios e o maior deles é encontrar profissionais qualificados (Anderson, 2011). A área de TI sofre um apagão de mão de obra no Brasil e esse problema reflete a baixa capacidade e produtividade das empresas do setor. Segundo relatório da Brasscom, Associação das Empresas de Tecnologia da Informação e Comunicação [TIC] e de Tecnologias Digitais, é estimado que 797 mil profissionais de TIC serão demandados até 2025, e o Brasil tem um déficit anual de 106 mil talentos (BRASSCOM, 2020).

Inserida nesse contexto, com diversos clientes do setor financeiro, uma grande quantidade de itens de trabalho em espera ("backlog") e com uma capacidade limitada de profissionais, uma empresa de Tech Banking de médio porte do interior de São Paulo têm experimentado a adoção de sistema puxado como o método "Kanban". Neste método, um item do "backlog" só é puxado para o desenvolvimento quando o time tem capacidade para lidar com ele. O item do "backlog" é representado por um "kanban" (cartão), e a necessidade do usuário é descrita como uma história anexada ao cartão (história do usuário). O time de desenvolvimento trabalha em ciclos curtos ("timebox"), de uma a quatro semanas, e tem a premissa de entregar funcionalidades com alta qualidade ao fim desse período. Um aspecto muito positivo do sistema puxado é a possibilidade de dar um ritmo sustentável para a equipe, através da limitação do "Work in Progress" (trabalho em progresso) [WIP]. O WIP impede que o colaborador tenha uma sobrecarga de trabalho (Anderson, 2011).

A excelência operacional é um dos objetivos estratégicos da empresa, com o cumprimento de 90% das datas de entrega como resultado-chave. No entanto, o time responsável pelo produto Conta Corrente enfrenta desafios no cumprimento de prazos e na priorização do backlog para atender diversos clientes. O prazo não deveria ser a principal preocupação do time. As iterações têm um prazo fixo de um mês para novas versões e uma semana para correções de erros. O desafio real é determinar o que priorizar e quantos itens do backlog o time pode lidar em um ciclo. Essa previsão e priorização são cruciais para negociações com os clientes e estratégias de mercado. A priorização idealmente deve ser feita pelo negócio, mas depende da capacidade do time de desenvolvimento para execução. A previsibilidade do que pode ser feito dentro do ciclo é essencial para a eficácia da priorização (Anderson, 2011). Cohn (2005) sugere reduzir incertezas na demanda, revisar médias históricas

e considerar variáveis significativas, incluindo a história do usuário, que detalha aspectos técnicos, funcionais e fluxos de exceção que influenciam a complexidade e o prazo (Anderson, 2011).

O objetivo geral deste trabalho é implementar um modelo de aprendizado de máquina [ML], que apoie o time de desenvolvimento do Conta Corrente da Tech Banking na seleção e priorização dos itens de "backlog" possíveis de serem tratados em um ciclo de desenvolvimento. Dentre os objetivos específicos estão: encontrar o melhor modelo supervisionado que faça a estimação dos prazos das atividades; a priorização dos itens de "backlog"; desenvolver um algoritmo que faça a sugestão das demandas na qual o time tem a capacidade de executar em um ciclo, considerando um prazo máximo pré-definido.

## 2. REFERENCIAL TEÓRICO

Priorizar e estimar são fatores importantes e altamente críticos em projetos de desenvolvimento de software (Cohn, 2005). Anderson (2011) inclui a priorização e a redução de variabilidade para a melhoria de previsibilidade entre os seis passos de sua receita de sucesso, porém destaca que atacar a variabilidade é uma tarefa difícil. Cohn (2005) propõe a estimativa por analogia como uma técnica alternativa para ter uma maior assertividade. Esta abordagem compara características das tarefas de desenvolvimento implementadas e concluídas, considerando tamanho, seus requisitos funcionais e não funcionais (Cohn, 2005). Cohn (2005) relembra também o cone das incertezas, criado por Barry Boehm e aprimorado por Steve McConnell, que estrutura um método para medir as incertezas de um projeto conforme seu amadurecimento. Quanto mais informações sobre o projeto menores são as incertezas (Cohn, 2005). Em projetos de software que aplicam o método "Kankan", toda a informação é evoluída e detalhada na história do usuário. Considerando a técnica de estimativa por analogia, a aplicação de modelos de regressão com técnicas de processamento de linguagem natural [NLP] pode ser usada para a prever o prazo de entrega de uma demanda com base na série histórica. Um modelo de regressão tenta avaliar a relação entre variáveis dependentes com uma ou mais variáveis independentes (Fávero & Belfiore, 2017). Neste caso a variável dependente é o prazo e as variáveis que tentam explicar o prazo são a história do usuário, suas propriedades e os profissionais envolvidos para execução da demanda. Vajjala, Majumder, Gupta, Surana (2020)

explicam que o processamento de linguagem natural tem a capacidade de extrair recursos textuais estatisticamente interpretáveis e estes recursos podem ser usados para treinar modelos de regressão.

Anderson (2011) relata que é muito comum a área de negócio delegar a responsabilidade para a área de tecnologia à tarefa de priorizar. Porém a priorização tem fatores que são inerentes ao negócio e a equipe de desenvolvimento, que não possui uma visão holística e estratégica, não tem competência para esta atividade (Cohn, 2005). Cohn (2005) destaca que os fatores de priorização devem considerar aspectos financeiros, de custo, de capacitação, riscos envolvidos e capacidade produtiva. Considerando esses fatores a priorização é dependente da estimativa prévia (Cohn, 2005). Saaty e Vargas (2012) propõe uma metodologia que pode facilitar a priorização das atividades. Saaty e Vargas (2012) destacam que o "Analytic Hierarchy Process" (Processo Hierárquico Analítico) (AHP) é um método prático e simples para a tomada decisão, na qual seleciona a melhor alternativa considerando vários critérios. Nesta abordagem é necessário que o decisor defina pesos para cada critério considerando a escala de Saaty. A metodologia também impõe limitações na quantidade de critérios de priorização. Santos, Araújo, Gomes (2021) propõe uma melhoria no método AHP, o AHP Gaussiano, que tira a necessidade do decisor ter que definir os pesos dos critérios. Os pesos são determinados com base na média e desvio padrão entre cada critério, chamado de fator Gaussiano. Esta metodologia não impõe limitações na quantidade de critérios.

## 3 MÉTODO

Este estudo foi desenvolvido através do método exploratório de natureza aplicada (Leavy, 2017), utilizando como objeto de análise, o banco de dados da ferramenta de gestão de demandas [JIRA] do Time do Produto de Conta Corrente de uma Tech Banking da cidade de Campinas, no estado de São Paulo. O banco de dados inclui as demandas iniciadas e concluídas no ano de 2022. Os dados do JIRA foram extraídos com autorização da área jurídica da empresa. O JIRA é um software proprietário sob licença da Atlassian e fornece recursos para gerenciamento de projetos ágeis (Atlassian, 2023).

O processo para desenvolvimento da solução seguiu uma adaptação das etapas da metodologia “Team Data Science Process” (Processo de ciência de dados de equipe) [TDSP] desenvolvida pela Microsoft (2023), conforme o esquema apresentado na Figura 1.

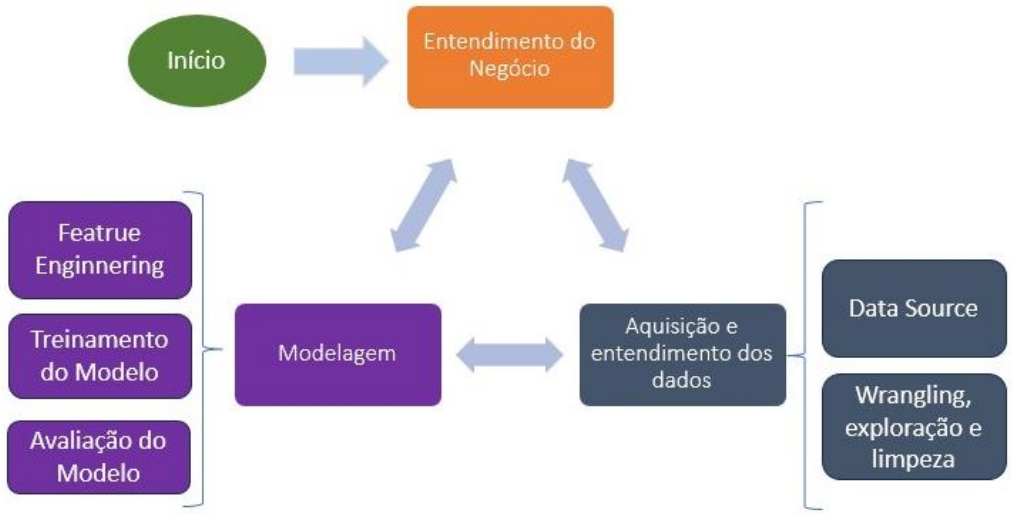


**Figura 1.** Representação da TDSP adaptado

Fonte: (Microsoft, 2023)

A metodologia prevê um ciclo iterativo e incremental de melhoria contínua e colaboração considerando as seguintes etapas:

a) Entendimento do Negócio
b) Aquisição de Dados e Entendimento dos dados
c) Modelagem

Na fase de Entendimento do Negócio, foi importante detalhar como eram os processos de desenvolvimento e suporte. A Tech Banking padronizou o método “Kanban” conforme esquema apresentado na Figura 2.

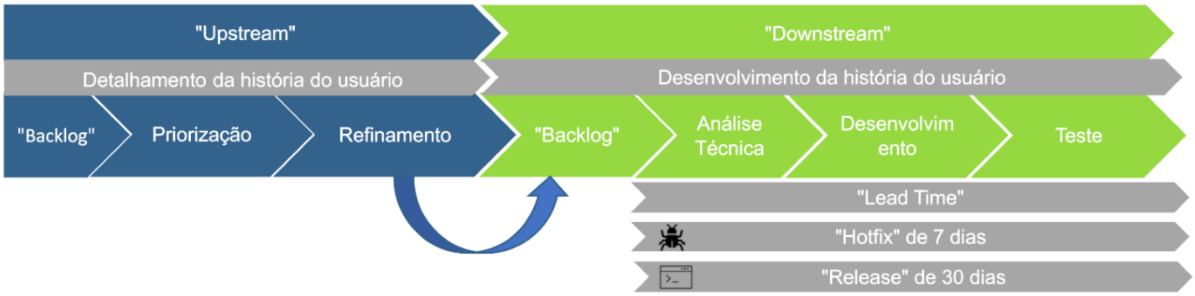


**Figura 2**. Método Kanban
Fonte: (Anderson, 2011)

O processo de desenvolvimento foi dividido em "Upstream" (montante) e "Downstream" (jusante). Na etapa de "Upstream" são consideradas as atividades de detalhamento das tarefas em backlog, a priorização, a triagem, o refinamento das demandas a serem desenvolvidas. A etapa de "Downstream" inclui as atividades de análise técnica, desenvolvimento e teste. O tempo percorrido no "Downstream" é chamado de "lead time" (tempo de ciclo) (Anderson, 2011). O "Upstream" constrói um "backlog" para o "Downstream". Além disso parte da equipe atende falhas e correções de erros, "hotfix", e a outra parte trabalha em melhorias e evoluções tecnológicas. A equipe que atende "hotfixes" tem ciclos de entregas de uma semana e a equipe de melhorias tem ciclos de entregas mensais. Este estudo considerou a etapa de "Downstream", onde o "lead time" é medido e a demanda em "backlog" tem o detalhamento necessário para o desenvolvimento.

A Aquisição dos Dados foi realizada através da Interface de Programação de Aplicativos com Transferência de Estado Representacional [API REST] disponibilizada pela ferramenta do JIRA que permite a consulta de dados através de comandos "JIRA Query Language" (Linguagem de Consulta do JIRA) [JQL], resultando em um conjunto de estruturas "JavaScript Object Notation" (Notação de Objeto Javascript) [JSON] com as "Issues" (Demandas) (ATLASSIAN, 2023). ATLASSIAN (2023) define "Issue" como uma demanda de qualquer natureza (e.g., Falha, Melhoria, Vulnerabilidade). Uma API REST é representada por um recurso do protocolo HTTP/HTTPS e pode ser executada através de verbos na língua inglesa (e.g. "GET", "POST", "PUT") (Munzert, Rubba, Meißner, Peter, 2015). Um recurso pode conter um ou mais parâmetros. A Tabela 1 mostra o recurso, o verbo e os parâmetros utilizados para a extração dos dados no JIRA.

**Tabela 1**
Recurso da API REST para a extração das “Issues”

| Verbo | Recurso | Parâmetro | Valor do Parâmetro |
|---|---|---|---|
| GET | /rest/api/2/search? | jql | "Equipe de Atuação" = "Core Conta" AND statusCategory in (Done) AND created >= '2022-01-01' AND resolutiondate <= '2022-12-31' ORDER BY assignee ASC, duedate ASC, created DESC |
| | | expand | changelog |

Fonte: (Atlassian, 2023)

O parâmetro “jql” recebe a consulta que foi realizada no JIRA com a linguagem JQL (Atlassian, 2023). A consulta considera apenas a equipe de atuação “Core Conta” (Time do produto de Conta Corrente) e as “Issues” prontas que foram criadas e finalizadas em 2022. O parâmetro “expand” (expansão) inclui campos adicionais que não viriam na consulta padrão (Atlassian, 2023). Neste caso a inclusão do campo “changelog” (registro de alterações) foi importante para esta análise pois possui dados sobre os históricos de alteração do status ciclo de vida das “Issues”. Os dados semiestruturados em JSON, gerado pela execução do verbo contra o recurso foi salvo em um arquivo nomeado como “ciclo-jira-conta-corrente.json”. Para melhor entendimento da estrutura de dados exportada do JIRA, foi criado um diagrama, detalhado na Figura 3, que representa a complexidade de uma “Issue”.

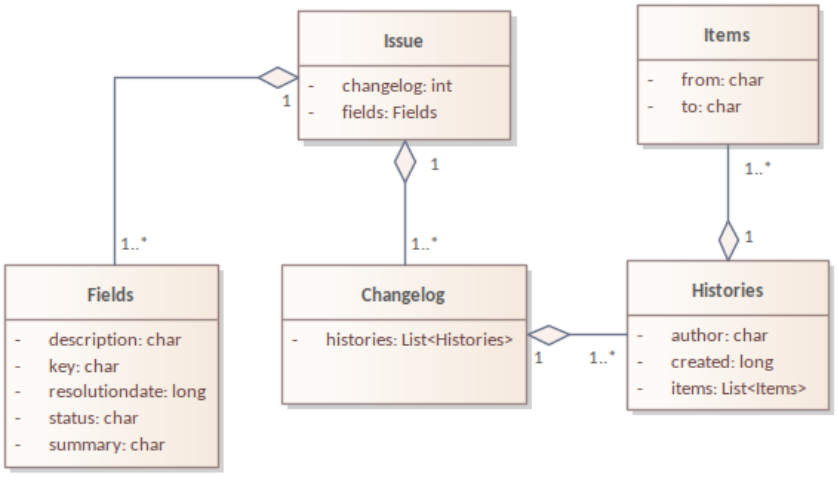


**Figura 3.** Representação da estrutura de dados de uma “Issue”

Fonte: (Atlassian, 2023)

Atlassian (2023) define que uma “Issue” possui “Fields” (Campos) que caracterizam uma demanda, como a descrição, o resumo, a prioridade, o responsável e outros. O “Changelog” contém os históricos (“Histories”) de alterações de qualquer atributo armazenado como itens alterados (“Items”) de uma “Issue”. O estado anterior do dado alterado é mantido no campo “from” e o novo estado é armazenado no campo “to” em “Items”. Em “Histories” (histórico) são armazenadas a data, hora e o autor da alteração realizada. Neste controle de troca de estados realizado em “Items”, foi possível identificar a transição entra as etapas de “Upstream” e “Downstream” conforme a diagrama de estado da Figura 4 (e.g. “from=Em Teste” e “to=Prontro para entrega”). Esta análise foi importante para identificar como capturar o momento em que a etapa de “Downstream” inicia, para o cálculo do “lead time”.

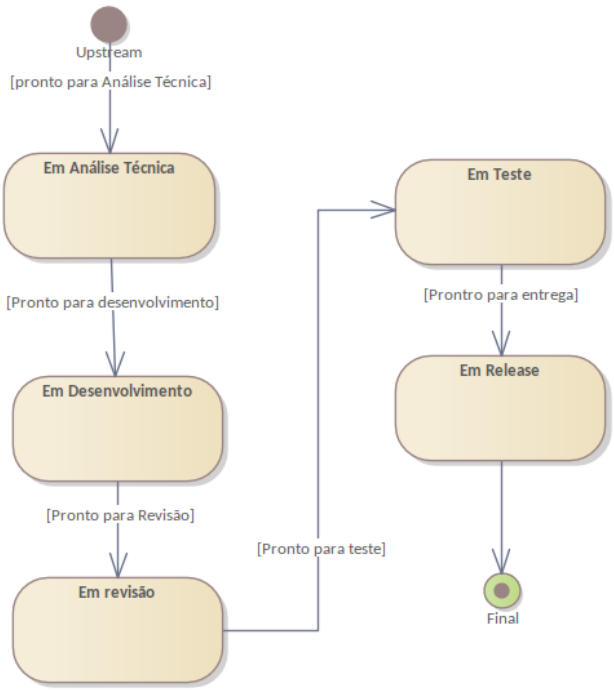


**Figura 4.** Diagrama de estado da transição das etapas de “Downstream”
Fonte: Dados originais da pesquisa

A etapa seguinte consistiu na transformação dos dados da estrutura JIRA para um “DataFrame” (quadro de dados) [DF]. A partir desta etapa todo o algoritmo do projeto foi desenvolvido com a linguagem R v.4.2.2, que possui um poderoso ferramental estatístico e uma comunidade ampla e ativa (Wickham et al., 2017). Levando em consideração a

complexidade da transformação e manipulação dos dados, foi criada uma biblioteca de algoritmos que auxiliaram nesta transformação, conforme detalhados na Tabela 2.

**Tabela 2**
Algoritmos auxiliares para a transformação dos dados

| Algoritmo | Descrição da função |
|---|---|
| Para transformação da estrutura básica da "Issue" em um DF | Este algoritmo realiza a transformação da estrutura básica de uma "Issue" em um DF no formato tabular. |
| Para interpretar o "changelog" | Este algoritmo procura o início e o fim da etapa de "Downstream", calcula o lead time e adiciona a coluna do "lead_time_hours" no DF |
| Para embaralhamento dos dados | Este algoritmo embaralha os dados sensíveis e não autorizados, e.g. nome de funcionários e clientes, para que os dados não sejam expostos na pesquisa. |
| Para aumentar o número de amostras | Este algoritmo aumenta o número de amostras do DF mantendo a características dos dados através de técnicas n-gramas e uso de sinônimos. |

Para a leitura e manipulação do "ciclo-jira-conta-corrente.json", assim como a manipulação e a transformação dos dados para o DF, foi necessário a utilização de algumas bibliotecas adicionais ao R. Funções da biblioteca "jsonlite" faz a leitura e manipulação do arquivo JSON (Ooms et al., 2022). A biblioteca "dplyr" faz a manipulação e a transformação dos dados para o DF (Wickham et al., 2022). A biblioteca "lubridate" auxilia na manipulação de datas e períodos (Spinu et al., 2022). A Biblioteca "rvest" faz a interpretação de páginas HTML (Wickham, 2022). O algoritmo de transformação da estrutura básica também remove as "Issues" canceladas ou descartadas. O algoritmo para interpretar o "changelog" determina o "lead time" de cada demanda, conforme o cálculo form. (1)

$$\boldsymbol{T = \big((t_1 - t_0) - (h_w + h_o)\big)h_u} \quad (1)$$

onde, T: é o "lead time" de uma "Issue" no "Downstream"; $t_1$: é a data e hora de encerramento da "Issue" no "Downstream"; $t_0$: é a data e hora de início da "Issue" no "Downstream"; $h_o$: é o

total de horas de feriados; $h_w$: total de horas de finais de semana; $h_u$: quantidade de horas de trabalho diário. No algoritmo foi considerado um período de oito horas de trabalho diário em $h_u$ e os feriados municipais do município de Campinas, armazenados no arquivo "holidays.csv". Após a transformação foi identificado que no período extraído, o time do Produto Conta Corrente nem sempre seguia todas as etapas do processo, conforme analisado na tabela 3. Foi observado que muitas "Issues" já saíam da fase de análise para a fase de teste, sem o registro da etapa do desenvolvimento.

**Tabela 3**
Quantidade de "Issues" por fase do ciclo de vida do "Downstream"

| Fase do Ciclo de Vida | Quantidade de "Issues" |
|---|---|
| Para Análise | 316 |
| Para Desenvolvimento | 107 |
| Para Teste | 338 |

Este cenário diminuiu bastante a amostra, considerando que o estudo é baseado no cumprimento do processo definido na fase de "Downstream". Vajjala et al. (2020) sugerem que se houver a necessidade de uma amostra maior, caso não tenha acesso a uma nova amostragem, é possível utilizar a técnica "Data Augmentation" (Aumento de Dados). A técnica aplicada deve garantir que os dados aumentados mantenham a mesma característica dos dados originais da amostra.

No DF criado, as variáveis "summary" (resumo) e "description" (descrição) representam a história do usuário na "Issue". A coluna "summary" é um resumo da história do usuário e a coluna "description" descreve com detalhes a necessidade da Issue. Como as duas colunas complementam a história de usuário, foi criado a coluna "user_story" (história do usuário) que une os textos das duas colunas no DF. Após essa transformação, os dados foram aumentados através da aplicação das técnicas de "n-grama" e substituição por sinônimos na coluna "user_story". Nas novas amostras, foram mantidas as mesmas variáveis, um "user_story" modificado e um novo identificador [KEY]. Vajjala et al. (2020) cita que o embaralhamento dos dados por "n-gramas" e a substituição das palavras por sinônimos, mantém a característica e a confiabilidade dos dados aumentados para análise. Para a

substituição dos sinônimos, foram usados dados extraídos da página de SINONIMOS (2011). Após o aumento dos dados, o DF foi salvo no arquivo “cc-2022-augmentation.csv”.

Antes de iniciar a escolha do melhor modelo para a forma funcional dos dados extraídos, foi necessário o pré-processamento dos textos das histórias de usuário. O pré-processamento é a etapa da NLP em que os textos são divididos em partes menores, depois são tratados, limpos e padronizados. Esse processo permite que a máquina encontre padrões textuais de forma mais eficiente (Vajjala et al., 2020). As etapas do pré-processamento estão descritas na Tabela 4.

**Tabela 4**
Etapas realizadas no pré-processamento

| Etapa | Descrição |
|---|---|
| Transformação em Caixa Baixa (“tolower”) | Transformação de todas as palavras da história de usuário em caixa baixa |
| Remoção de palavras irrelevantes (“stopwords”) | Remoção das palavras irrelevantes para a análise do modelo (e.g. de, para, até) |
| Remoção de palavras reservadas e não autorizadas para a análise (“tech stopwords”) | Remoção de palavras técnicas irrelevantes e não autorizadas para análise (e.g. nome da empresa, domínios internos) |
| Remoção de números | Neste cenário, os números não fazem sentido para a análise textual |
| Remoção de pontuação | As pontuações não têm significado para uma análise de padrões de complexidade. |
| Remoção de espaços em branco | Espaços em branco não são considerados na análise |
| Redução das palavras ao seu tronco (“stem”) | Redução das palavras flexionadas ao seu tronco. |
| Divisão dos Textos em “bi-gramas" | Quebra dos textos em pares de palavras, onde cada par vira uma coluna no DF, e.g. “amanhã vai”. |

Fonte: (Vajjala et al., 2020)

Para realizar a etapa de pré-processamento foi utilizado a função “tm_map” da biblioteca “tm”. A biblioteca “tm” tem a infraestrutura necessária para realizar toda a etapa de pré-processamento. (Feinerer et al., 2008).

Concluindo as fases de Entendimento de Negócio e Aquisição de Dados, iniciou-se a fase de Modelagem. Esta fase incluiu a etapa de “Feature Engineering” (Engenharia de recursos), cujo objetivo principal foi transformar as variáveis explicativas selecionadas em um

vetor numérico para que os algoritmos de aprendizado de máquina possam entender, ou seja, transformar variáveis categóricas em variáveis “dummy” (Vajjala et al., 2020). Fávero e Belfiore (2017) explicam que as variáveis “dummy” são representações binárias das variáveis categóricas, onde 0 representa a ausência da variável e 1 indica a presença da variável na interpretação do algoritmo de ML. A Tabela 5 lista as variáveis explicativas categóricas selecionadas que passaram pelo processo de transformação em variáveis “dummy”.

**Tabela 5**
Variáveis categóricas explicativas transformadas em variáveis dammy

| Variável | Tipo de dado no DF | Descrição |
|---|---|---|
| “developer” | chr | Representa o profissional responsável pelo desenvolvimento do software |
| “tester” | chr | Representa o profissional responsável pelo teste e pela qualidade final do software |
| “customer” | chr | Representa o cliente solicitante ou impactado pela demanda |
| “priority” | chr | Representa o nível de prioridade da demanda. Pode ser “Baixa”, “Media”, “Alta” e “Urgente” |
| “issuetype” | chr | Representa o tipo de demanda. (e.g. "Performance", "Falha", "Tarefa", "Vulnerabilidade de Segurança", “Melhoria”) |
| “projectkey” | chr | Representa o módulo do sistema na qual a demanda afeta. O time do Produto do Conta Corrente possui dois módulos: “CC” e “TS” |
| “category” | chr | Representa a categoria que agrupa os tipos de demandas (“issuetype”) em um segundo nível de agrupamentos. |
| “user_story” | chr | Representa a história de usuário. Esta variável representa as diversas variáveis geradas na etapa de pré-processamento (divisão de textos em bi-gramas), na qual fica inviável listar nesta tabela. |

As variáveis explicativas listadas na Tabela 5 são as variáveis selecionadas para análise inicial do modelo, cujo objetivo é explicar o “lead time” do “Downstream”. Estas são o máximo de variáveis disponíveis antes do início do “Downstream”. É importante entender que as variáveis “developer” e “tester” representam o fator humano. Outro aspecto importante e que pode afetar assertividade da previsão, é a qualidade da história do usuário. Histórias menos

detalhadas reduzem a possibilidade de o modelo encontrar padrões que expliquem o "lead time", tendo em vista a limitação de variáveis explicativas ao final do "Upstream".

O próximo passo foi a escolha do modelo de distribuição. Fávero e Belfiore (2017) destacam que quanto mais a distribuição se ajustar a forma funcional, melhor será a capacidade do modelo em fazer previsões. Também foram escolhidas as estatísticas $R^2$, Log de Máxima Verossimilhança [Loglike], "Akaike Information Criterion" (Critério de informação Akaike) [AIC] e "Bayesian Information Criterion" (Critério Bayesiano de Informação) [BIC] como critérios de escolha do modelo. Fávero e Belfiore (2017) explicam que o $R^2$ mede a capacidade de um modelo de regressão explicar uma variável explicada Y. Freedman (2009) destaca que o Loglike representa a probabilidade dos dados observados com base nos parâmetros do modelo estatístico. AIC e BIC são medidas estatísticas que penalizam a quantidade de parâmetros do modelo, ou seja, penalizam modelos mais complexos (Bishop, 2006).

As etapas utilizadas para a escolha do modelo de distribuição candidata estão listadas na tabela 6.

**Tabela 6**
Etapas para escolha do modelo de distribuição candidata

| Etapa | Descrição |
|---|---|
| Análise da Assimetria e da Curtose | Análise empírica do peso da cauda e da assimetria do modelo (Gráfico de Cullen e Frey) |
| Ajuste da amostra com as distribuições identificadas na análise empírica | Utilização da função "fitdist" do pacote "fitdistrplus" para análise empírica de Loglike, AIC e BIC de diferentes distribuições. |
| Escolha da distribuição candidata | Escolha da distribuição candidata |

Fonte: (Delignette-Muller e Dutang, 2015)

Uma etapa prévia para a escolha da distribuição foi realizada com funções do pacote "fitdistrplus". O pacote contém funções para análise empírica de modelos de distribuição (Delignette-Muller e Dutang, 2015). Isso permite um direcionamento inicial, porém os modelos devem ser aplicados e medidos através de técnicas de validação cruzada. A função "descdist" foi utilizada para a geração das estatísticas básica e a análise da Assimetria e Curtose através

do gráfico de Cullen e Frey (Delignette-Muller & Dutang, 2015). A função "fitdist" foi utilizada para análise empírica das diferentes distribuições direcionadas pela análise de Assimetria e Curtose (Delignette-Muller & Dutang, 2015).

Sobre a escolha do algoritmo de ML, para a realização da previsão do "lead time", o estudo direciona a utilização de um modelo de regressão do tipo "Generalized Linear Model" (Modelo Linear Generalizado) [GLM], pois os resíduos se apresentaram diferente da distribuição normal (Fávero & Belfiore, 2017).

Fávero e Belfiore (2017) destacam que nos modelos de regressão, é importante manter apenas as variáveis estatisticamente significantes para o nível de confiança determinado, com base na análise da estatística t de "Student". Essa análise e ajuste, pode ser automatizado através do procedimento de "stepwise" (passo a passo), que exclui automaticamente as variáveis sem significância estatística.

Como o processo de NLP gerou amostras com alta dimensionalidade, métodos tradicionais de "stepwise" tiveram problemas de performance. Optou-se pelo uso de algoritmos que utilizam a abordagem da penalidade. Então foram selecionados os algoritmos GLMNET e o XGBoost para a comparação. A biblioteca do R "glmnet" implementa um algoritmo de ML que penaliza as variáveis pela análise de máxima verossimilhança, utilizando a regularização LASSO (Laço) [L1], Ridge (Cume) [L2] ou "Elastic Net" (Rede Elástica) [L1/L2] (Friedman et al., 2023). A regressão LASSO penaliza as variáveis menos importantes, tendendo-os a zero, fazendo com que as variáveis sejam eliminadas. A regressão Ridge reduz a sensibilidade dos coeficientes fazendo pequenas mudanças nos dados. A regressão "Elastic Net" faz um equilíbrio entre a L1 e L2. (Friedman et al., 2010). GLMNET revolve regressões GLM conforme a fórmula geral do form. (2) (Friedman et al., 2010):

$$\min_{\beta_0,\beta} \frac{1}{n}\sum_{i=1}^{n} w_i\, l_i(y_i,\ \beta_0 + \beta^T x_i) + \lambda\left[\frac{1-\alpha}{2}||\beta||\frac{2}{2} + \alpha||\beta||_1\right] \quad (2)$$

onde, $Y_i$: é variável explicada da observação i; $X_i$: é o vetor das variáveis explicativas da observação i; β: é o vetor de coeficientes a serem estimados para cada variável explicativa, $\beta_0$: é o intercepto do hiperplano. n: é o número de observações. $W_i$: é o peso associado a observação

i na função de perda. $l_i$: é a função de perda usada para medir o resíduo da observação i e.g "deviance". λ: é o hiper parâmetro que controla o grau de penalização dos coeficientes. α:é o hiper parâmetro que controla a proporção entre a regularização L1 e L2.

O "Extreme Gradient Boosting" (Aumento de Gradiente Extremo) [XGBoost], é um algoritmo de ML, de alta performance, baseado em árvores de decisão. Tem a capacidade de tratar amostras esparsas, com alta dimensionalidade e não lineares. Ele cria uma cadeia de modelos e realiza um processo de melhoria contínua, onde cada novo modelo otimiza a previsão do modelo anterior. XGBoost também permite o uso de estratégias de penalização L1 e L2 (Chen & Guestrin, 2016). A biblioteca do R "xgboost" implementa uma infraestrutura para a estratégia XGBoost (XGBoost, 2022). Nela, contém algoritmos que permitem a validação cruzada para escolhas de hiper parâmetros, treinamento e medição de modelos XGBoost (XGBoost, 2022). XGBoost tem a função objetivo conforme form. (3) (Chen & Guestrin, 2016):

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \tag{3}$$

onde, $\mathcal{L}^{(t)}$: é a função perda para a iteração t. n: número de amostras; l: função perda pra a observação i. $y_i$: é a variável explicada original; $\hat{y}_i^{(t-1)}$: é a variável predita na iteração anterior do modelo; $f_t$: é a função de previsão do modelo na iteração atual. $X_i$: é a variável explicativa da observação i. $\Omega(f_t)$: é a penalidade aplicada para a complexidade do modelo na iteração t.
As etapas que foram necessárias para a construção dos modelos de ML estão listadas na Tabela 7.

**Tabela 7**
Etapas para construção dos modelos de ML

| Etapa | Descrição |
|---|---|
| Separação das amostras em grupos de Treino, Teste, Validação e separação das variáveis explicativas e a variável explicada “lead_time_hours” para cada base. | As amostras foram separadas para avaliar se o modelo é capaz de generalizar para a previsão de “lead_time_hours” com diferentes amostras e se o modelo não está com sobre ajuste. A amostra de validação foi utilizada para avaliação do algoritmo de sugestão. |
| Validação cruzada para escolha de hiper parâmetros | Foram utilizadas técnicas de “Grid Search” (Pesquisa em grade) para encontrar os melhores hiper parâmetros para os algoritmos avaliados |
| Construção e treinamento do modelo | Treinamento dos modelos considerando os hiper parâmetros encontrados e as amostras de treinamento |
| Teste do modelo treinado | Teste do modelo treinado com o grupo de amostras de teste. |
| Registro das estatísticas | Registro das estatísticas $R^2$, AIC, BIC e Loglike |

Fonte: (Microsoft, 2023)

Na primeira etapa, a lista de nomes das variáveis explicativas da amostra de treino foi salva em um arquivo “terms.csv”. Este arquivo serviu como um dicionário para o ajuste dos parâmetros de entrada do modelo, na etapa de validação. Com os modelos GLMNET e XGBoost construídos, a escolha foi com base na avaliação das estatísticas $R^2$, AIC, BIC e Loglike.

Por fim foi criado o algoritmo que faz a sugestão de demandas, considerando a capacidade do time, dentro de um limite de dias úteis, pré-definido. O algoritmo recebe 4 parâmetros de entrada: um DF, das demandas em “backlog”, com as colunas “key”, “priority”, “issuetype”, “projectkey”, “category”, “summary” e “description”; um DF com a lista de desenvolvedores da equipe; um DF com a lista de testadores da equipe; o tempo em dias do ciclo. O algoritmo então segue os passos da Tabela 8.

**Tabela 8**
Passos do algoritmo que faz a sugestão das demandas

| Passo | Descrição |
|---|---|
| Ranqueamento das demandas por critério de priorização | Foi aplicado o Método AHP Gaussiano para criar um ranque de demandas. Nesta etapa o algoritmo cria uma coluna “ranking” que indica a prioridade das demandas. São considerados critérios carregados dos arquivos “customer_weights.csv”, “issuetype_weights.csv” e “priority_weights.csv”. |
| Combinação de todas as possibilidades entre testador e desenvolvedor para executar uma demanda | Foi criado um DF com todas as combinações possíveis entre desenvolvedor e testador para o atendimento da demanda |
| Previsão do “lead_time_hours” para todas as combinações criadas | Foi utilizado o modelo de ML escolhido para prever o “lead_time_hours” de todas as combinações |
| Filtro das demandas combinadas pelo menor “lead_time_hours” estimado. | Dentre todas as combinações, o algoritmo filtra o que tem menor “lead_time_hours” previsto de cada demanda. |
| Criação da lista de sugestão de quebra de história de usuário | As demandas com o “lead_time_hours” previsto maior que o tempo definido para o ciclo são inseridas numa lista de sugestões para a quebra em demandas menores. |
| Criação da lista de sugestão de demandas para o ciclo | São mantidos apenas as demandas com “lead_time_hours” menor que o tempo definido pelo ciclo, ordenadas pelo ranking. Também é considerado o tempo máximo da capacidade dos recursos (desenvolvedor e testador), e.g. 8 horas por dia no período do ciclo. |

O Ranqueamento das demandas aplica o Método AHP Gaussiano conforme os passos da Figura 5.

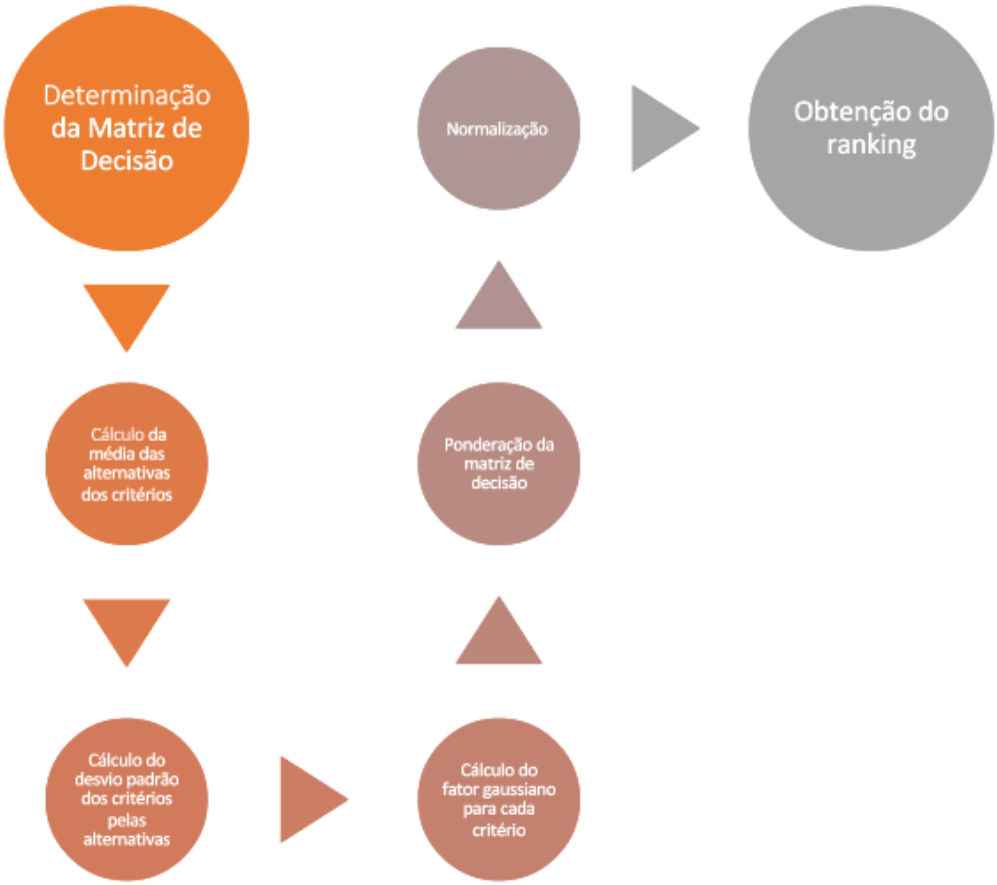


**Figura 5**. Método AHP Gaussiano
Fonte: (Santos et al., 2021)

Neste estudo foram utilizados como critérios de priorização, o cliente, o tipo de demanda e a prioridade. Com este conjunto de critérios, o uso do AHP clássico atenderia através da aplicação de pesos da escala Saaty. Porém o uso do método AHP Gaussiano trará mais flexibilidade na inclusão de novos critérios conforme a evolução deste projeto para uma escala maior.

## 4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

O gráfico de Cullen e Frey da Figura 6, indicou uma distribuição com características próximas da distribuição Gama e Lognormal.

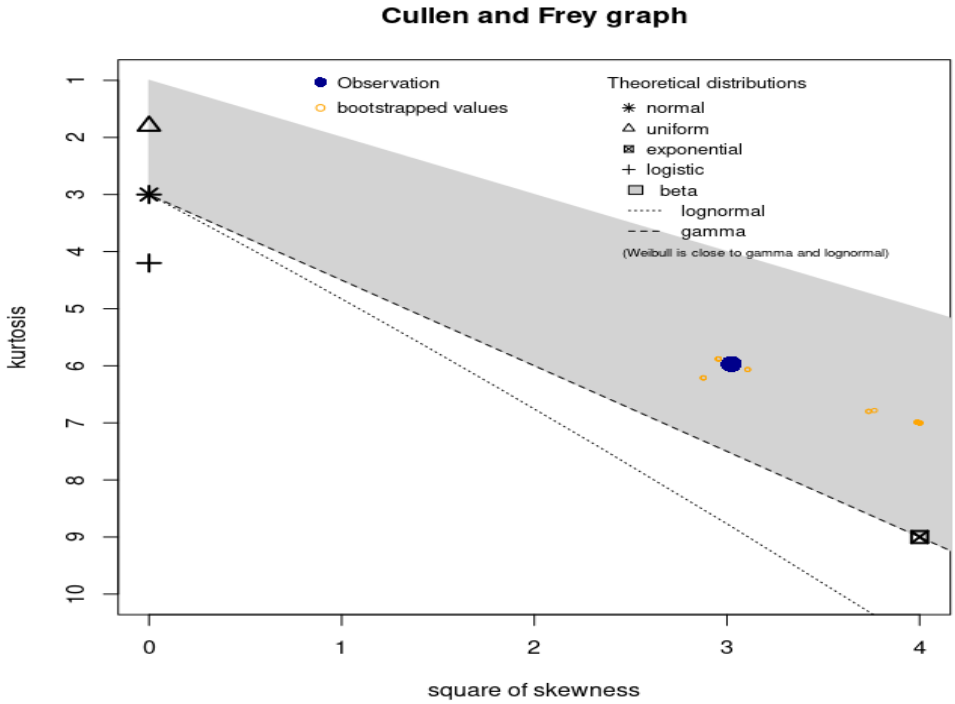


**Figura 6**. Gráfico de Cullen e Frey

As estatísticas listadas na Tabela 9, mostraram uma faixa considerável de variação, observados os valores de máximo e mínimo. Pela mediana ser menor que a média, os dados apresentaram uma Assimetria positiva. Considerando um desvio padrão maior que a média, foi possível observar uma grande dispersão dos dados. A Assimetria positiva confirmou a presença de uma calda mais longa a direita. A Curtose inferior a 6 indicou que os dados têm uma distribuição menos concentrada em torno da média, com caudas mais leves e um pico menos acentuado do que o de uma distribuição normal.

**Tabela 9**
Resumo das estatísticas

| Resumo das estatísticas | | | |
|---|---|---|---|
| Mínimo: | 5.448788 | Máximo: | 1276.102 |
| Mediana: | 163.9838 | | |
| Média: | 402.3796 | | |
| Desvio padrão estimado: | 596.5805 | | |
| Assimetria estimada: | 1.738421 | | |
| Curtose estimada: | 5.970239 | | |

Na análise dos dados, foi realizado um ajuste da variável explicada “lead_time_hours” contra as distribuições Gama e Lognormal. Os indicadores de Loglike, AIC e BIC listados na Tabela 10, indicam que a distribuição Gama possui um ajuste melhor aos dados. Fávero e Belfiore (2017) explicam que Gama é uma das distribuições de probabilidade, para variáveis aleatórias contínuas, mais gerais dentre as demais e amplamente aplicada em diversas áreas de estudo.

**Tabela 10**
Comparação das estimativas de Loglikelihood, AIC e BIC

| Distribuição Gama | | Distribuição Lognormal | |
|---|---|---|---|
| Loglikelihood | -7943.583 | Loglikelihood | -8430.359 |
| AIC | 15891.17 | AIC | 16864.72 |
| BIC | 15901.63 | BIC | 16875.19 |

Na Figura 7, o histograma do “leadtime” com as curvas teóricas das distribuições Gama e Lognormal mostram um ajuste melhor em Gama.

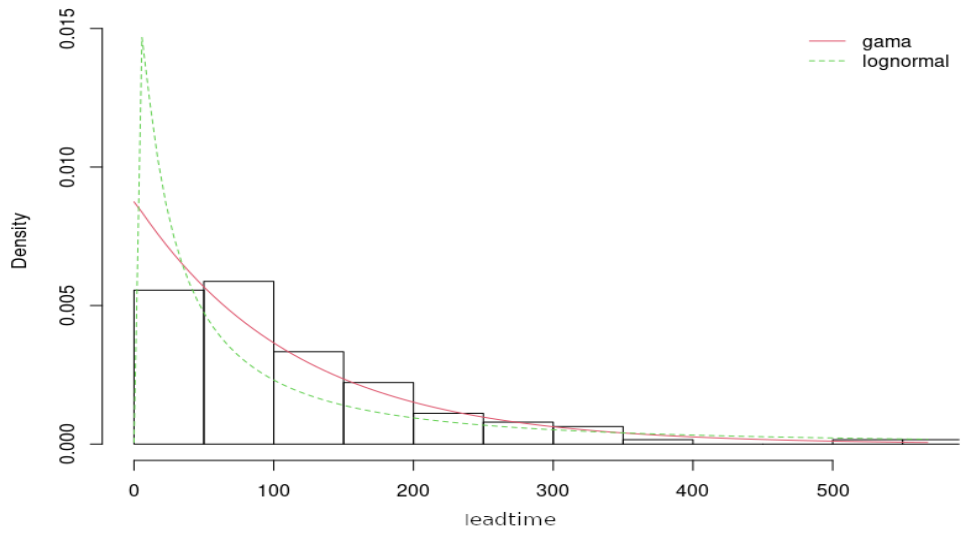


**Figura 7.** Histograma do “leadtime” ajustados nas distribuições teóricas de Gama e Lognormal

A análise indicou uma distribuição Gama, porém foi observado no gráfico da Figura 6 que os dados possuem diferentes características de dispersão e assimetria. Considerando essas observações adicionais, além da distribuição Gama, também foi selecionada a distribuição Tweetie para análise dos modelos de ML. Dunn e Smyth (2018) destacam que a distribuição da família Tweetie engloba várias outras distribuições dentro do Modelos de Dispersão Exponencial [MDE] e são úteis na modelagem de dados com diferentes características de dispersão e assimetria.

As distribuições Gama e Tweetie formam aplicadas inicialmente no algoritmo de ML XGBoost. No R foi utilizada a função “xgb.cv” da biblioteca “xgboost” para a realização da validação cruzada [VC], com o objetivo de encontrar os melhores hiper parâmetros para o modelo (XGBoost, 2022). Os parâmetros utilizados para iniciar a VC de cada distribuição estão listados na Tabela 11.

**Tabela 11**
Parâmetros utilizados na validação cruzada (“xgb.cv”) do algoritmo XGBoost para cada classe de distribuição.

| Distribuição Gama | | Distribuição Tweetie | |
|---|---|---|---|
| Parâmetro | Valor | Parâmetro | Valor |
| “objective” | “reg:gamma” | “objective” | “reg:tweedie” |
| “booster” | “gbtree" | “booster” | “gbtree" |
| “nthread” | 5 | “nthread” | 5 |
| “max_depth” | 7 | “max_depth” | 7 |
| “eval_metric” | “rmse” | “eval_metric” | “rmse” |
| “nrounds” | 100 | “nrounds” | 100 |
| “nfold” | 10 | “nfold” | 10 |
| “early_stopping_rounds” | 10 | “early_stopping_rounds” | 10 |
| | | “tweedie_variance_power” | 1.5 |

Estes são os hiperparâmetros considerados ótimos, escolhidos por meio de experimentos com diversas VC's, onde foi observada a capacidade do modelo treinado de generalizar, usando as amostras de teste. Foram também considerados os indicadores estatísticos Loglike, AIC e BIC. Na Tabela 11, o parâmetro "objective" define a classe de distribuição usada para otimizar o modelo no treinamento; "booster" define a estratégia da arquitetura utilizada (neste caso,

"gbtree" define uma estratégia de árvore de decisão); "nthread" define quantas instâncias do modelo foram treinadas em paralelo; "max_depth" define a profundidade da árvore de decisão; "eval_metric" define a métrica utilizada para escolher o modelo. "nrounds" define quantas iterações o modelo realizou para obter uma otimização máxima; "early_stopping_rounds" define quantas iterações o modelo continuou a mais sem ter um ganho de desempenho e parou. "nfold" define o número de iterações de VC's que foram realizadas (XGBoost, 2022). No caso da distribuição Tweedie, foi necessário incluir o parâmetro "tweedie_variance_power" que representa o $\xi$ da função de variação de potência (Dunn e Smyth, 2018).

Após a conclusão da etapa de VC, a função "xgb.cv" retornou à iteração ("best_iteration") que teve o melhor desempenho. Então, o modelo do algoritmo XGBoost foi treinado com as amostras de treinamento usando a função "xgb.train", considerando o "best_iteration" da VC. Após o treinamento, os modelos foram testados com as amostras de teste. A tabela 12 lista os indicadores estatísticos dos modelos testados com o algoritmo XGBoost e as distribuições Gama e Tweedie.

**Tabela 12**
Indicadores estatísticos do algoritmo XGBoost aplicados na amostra de teste.

| Distribuição Gama | | Distribuição Tweetie | |
|---|---|---|---|
| | Indicador | | Indicador |
| Loglikelihood | -1576.136 | Loglikelihood | -1570.522 |
| $R^2$ | 0.6733450 | $R^2$ | 0.7499678 |
| AIC | 207490.3 | AIC | 207479 |
| BIC | 1756209 | BIC | 1756198 |

Levando em consideração os indicadores de complexidade e probabilidade, a distribuição Tweetie teve uma melhor performance em relação a distribuição Gama.

O algoritmo GLMNET não implementa a classe de distribuição Tweetie. Porém, como os indicadores de Loglike, AIC e BIC da distribuição Gama, na implementação XGBoost se apresentaram muito próximos do modelo da distribuição Tweetie, foi treinado um modelo GLMNET com distribuição Gama para comparação. No R foi utilizada a função “cv.glmnet”

para a VC do modelo (Friedman et al., 2010). Os parâmetros utilizados na função para iniciar a VC do modelo estão listados na Tabela 13.

**Tabela 13**
Parâmetros utilizados na validação cruzada (“cv.glmnet”) do algoritmo GLMNET.

| Distribuição Gama | |
|---|---|
| Parâmetro | Valor |
| “nfolds" | 10 |
| “family" | “Gamma ()” |
| type.measure | "mse" |

A VC da GLMNET foi realizada em duas etapas. Primeiro foi executado a VC com 10 iterações (“nfolds”), ajustado na distribuição Gama (e.g. “family” = “Gamma ()”). Esta VC encontrou o menor valor possível de λ. Com o valor do λ mínimo, foi realizado um loop (laço ou “Grid Search”) com o valor de α variando de entre 0 e 1. A cada variação de α foi treinado um modelo usando a função “glmnet” com os valores de λ mínimo e a variação de α. Então foi escolhido o α onde o “glmnet” gerava o menor erro quadrático médio. Com os valores ótimos de λ e α encontrados, o modelo foi treinado com as amostras de treinamento. Após o treinamento o modelo foi testado com as amostras de teste. A Tabela 14 mostra a comparação dos indicadores obtidos no teste do modelo GLMNET com o modelo XGBoost.

**Tabela 14**
Comparação das estatísticas dos modelos em GLMNET e XGBoost

| Distribuição Gama GLMNET | | Distribuição Tweetie XGBoost | |
|---|---|---|---|
| | Indicador | | Indicador |
| Loglikelihood | -1490.1 | Loglikelihood | -1570.522 |
| $R^2$ | 0.584895 | $R^2$ | 0.7499678 |
| AIC | 207318.2 | AIC | 207479 |
| BIC | 1756037 | BIC | 1756198 |

Comparando os indicadores de Loglikelihood, AIC e BIC que estatisticamente são muito próximos e por se tratar de uma distribuição não linear, o XGBoost apresentou um $R^2$ mais alto, e foi selecionado como melhor algoritmo para a previsão dos prazos.

A etapa final deste estudo iniciou com a criação de um cenário hipotético, criado com dados reais, que inclui uma lista de “backlog” de 22 demandas e um time com 6 desenvolvedores e 3 testadores. Os desenvolvedores e testadores tem um expediente de 8 horas diária. Este cenário foi montado com parte das amostras separadas para a validação e com a escolha aleatória de desenvolvedores e testadores da amostra total. Neste cenário, o ciclo completo de desenvolvimento (“Downstream”), deveria ocorrer em 15 dias úteis (120 horas). Foi aplicado o modelo XGBoost da distribuição Tweetie na previsão do “lead_time_hours”. O algoritmo então recebeu um DF com as demandas em “backlog”, um DF com os desenvolvedores, um DF com os testadores e o número de dias máximo do ciclo. A Tabela 15 apresenta as sugestões realizadas pelo algoritmo, listando as demandas priorizadas e possíveis de serem realizadas com base na capacidade do time para um ciclo de 15 dias.

**Tabela 15**
Sugestões realizadas pelo algoritmo para o próximo ciclo de 15 dias

| Demandas Sugeridas | | | | | Demandas para melhoria | |
|---|---|---|---|---|---|---|
| “Ranking” | “Issue” | Desenv. | Testador | “Lead Time” | “Issue” | “Lead Time” |
| 0.06328321 | CC-2890 | jr | rMa | 8 | CC-3624 | 141 |
| 0.05756873 | CC-3278 | et | ula | 48 | CC-3617 | 129 |
| 0.05582868 | TS-450 | gea | dro | 2 | CC-3618 | 137 |
| 0.05244913 | CC-3024 | im | rMa | 53 | CC-2782 | 147 |
| 0.04586463 | CC-2561 | wj | ula | 24 | CC-3666 | 148 |
| 0.04439973 | CC-3437 | vp | dro | 39 | | |
| 0.04110747 | CC-2682 | jr | rMa | 49 | | |
| 0.03598787 | CC-3340 | et | ula | 18 | | |
| 0.02940337 | CC-3557 | gea | dro | 23 | | |

Na Tabela 15, a coluna “Demandas Sugeridas” apresenta a lista de demandas priorizadas para o ciclo, indicando a combinação do desenvolvedor e o testador que levam o tempo ótimo para realizar a execução. Na coluna “Demandas para melhoria”, são listadas as demandas cujo a previsão realizada para o “lead time” foi maior do que o tempo definido para o ciclo. Neste caso é sugerido a quebra da história do usuário em partes menores.

## 3 CONCLUSÃO

Quando há muitas variáveis independentes na amostra, algoritmos de ML como GLMNET e XGBoost melhoram a precisão, reduzem o overfitting e simplificam a modelagem ao selecionar automaticamente as variáveis mais relevantes. No entanto, a regularização pode dificultar a interpretação dos coeficientes, e a sensibilidade dos hiperparâmetros é um ponto crítico. Além disso, há um custo computacional elevado. O próximo passo é aplicar modelagem multinível em diferentes contextos bancários. Este algoritmo é uma ferramenta de apoio e não substitui as decisões da equipe, mas pode aumentar a precisão na previsão de prazos, reduzindo pressões e melhorando a saúde mental dos colaboradores.

## REFERÊNCIAS

Anderson, D. J. (2011). Kaban, Mudança Evolucionária de Sucesso para Seu Negócio de Tecnologia. Blue Hole Press.

Atlassian. (2023a). What is an issue? https://support.atlassian.com/jira-software-cloud/docs/what-is-an-issue/.

Atlassian. (2023b). O que é o Jira Software? https://www.atlassian.com/br/software/jira/guides/getting-started.

Atlassian. (2023c). REST APIs. https://developer.atlassian.com/server/jira/platform/rest-apis/.

Atlassian. (2023d). REST API V2. https://developer.atlassian.com/cloud/jira/platform/rest/v2/intro/.

Atlassian. (2023e). Search for issues using JQL. https://developer.atlassian.com/cloud/jira/platform/rest/v2/api-group-issue-search/#api-rest-api-2-search-get.

Atlassian. (2023f). What is advanced searching in Jira Cloud? https://support.atlassian.com/jira-software-cloud/docs/what-is-advanced-searching-in-jira-cloud.

Associação das Empresas de Tecnologia da Informação e Comunicação (TIC) e de Tecnologias Digitais [BRASSCOM]. (2020). Estudo da Brasscom aponta demanda de 797 mil profissionais de tecnologia até 2025. Retrieved from https://brasscom.org.br/estudo-da-brasscom-aponta-demanda-de-797-mil-profissionais-de-tecnologia-ate-2025.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer Science+Business Media.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. https://dl.acm.org/doi/10.1145/2939672.2939785.

Cohn, M. (2005). Agile Estimating and Planning. Pearson Education.

Delignette-Muller, M. L., & Dutang, C. (2015). fitdistrplus: An R Package for Fitting Distributions. Retrieved from https://www.jstatsoft.org/article/view/v064i04.

Dunn, P. K., & Smyth, G. K. (2018). Generalized Linear Models With Examples in R. Springer Science+Business Media.
Fávero, L. P., & Belfiore, P. (2019). Data Science for Business and Decision Making. Elsevier Inc.
Fávero, L. P., & Belfiore, P. (2017). MANUAL DE ANÁLISE DE DADOS: Estatística e Modelagem Multivariada com Excel®, SPSS® e Stata®. Elsevier Editora Ltda.
Feinerer, I., Hornik, K., & Meyer, D. (2008). Text Mining Infrastructure in R [tm]. https://www.jstatsoft.org/article/view/v025i05.
Friedman, J., Hastie, T., & Tibshirani, R. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition. Springer Science+Business Media.
Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. https://www.jstatsoft.org/article/view/v033i01.
Freedman, D. A. (2009). Statistical Models: Theory and Practice. Cambridge University Press.
Friedman, J., Hastie, T., Tibshirani, R., Narasimmhan, B., Tay, K., Simon, N., & Yang, J. (2023). The family Argument for glmnet. https://glmnet.stanford.edu/articles/glmnetFamily.html.
Kuhn, M., & Johnson, K. (2013). Applied Predictive Modeling. Springer Science+Business Media.
Leavy, P. (2017). Research Design: Quantitative, Qualitative, Mixed Methods, Arts-Based, and Community-Based Participatory Research Approaches. The Guilford Press.
Meirelles, F. S. (2022). Panorama do Uso de TI no Brasil – 2022. https://portal.fgv.br/artigos/panorama-uso-ti-brasil-2022.
MICROSOFT. (2023). What is the Team Data Science Process? https://learn.microsoft.com/en-us/azure/architecture/data-science-process/overview.
Munzert, S., Rubba, C., Meißner, P., & Nyhuis, D. (2015). Automated Data Collection with R: A Practical Guide to Web Scraping and Text Mining. John Wiley & Sons Ltd.
Ooms, J., Lang, H., Temple, D., & Hilaiel, L. (2022). jsonlite. https://github.com/jeroen/jsonlite/.
Saaty, T. L., & Vargas, L. G. (2012). Models, Methods, Concepts & Applications of the Analytic Hierarchy Process, Second Edition. Springer Science+Business Media, LLC.
Santos, M. dos C., Costa, I. P. de A., & Gomes, S. C. F. (2021). Multicriteria decision-making in the selection of warships: a new approach to the AHP method. International Journal of the Analytic Hierarchy Process, 13(1), 833.
Spinu, V., Grolemund, G., & Wickham, H. (2022). lubridate. https://lubridate.tidyverse.org/.
Vajjala, S., Majumder, B., Gupta, A., & Surana, H. (2020). Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems. O'Reilly Media, Inc.
Wickham, H. (2022). rvest. from https://rvest.tidyverse.org/.
Wickham, H., Grolemund, G. (2017). R for Data Science: Import, Tidy, Transform, Visualize, and Model Data. O'Reilly Media, Inc.
Wickham, H., François, R., Henry, L., Müller, K., & Vaughan, D. (2022). dplyr. https://dplyr.tidyverse.org/.
XGBoost. (2022). XGBoost R Package. https://xgboost.readthedocs.io/en/latest/R-package/index.html.