UNIVERSIDADE FEDERAL DE OURO PRETO
ESCOLA DE MINAS
COLEGIADO DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO - CECAU

ALAN SOUZA SANTANDREA

MLOPS: INTRODUÇÃO AO TEMA E ESTUDO DE CASO

MONOGRAFIA DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Ouro Preto, 2022

ALAN SOUZA SANTANDREA

MLOPS: INTRODUÇÃO AO TEMA E ESTUDO DE CASO

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos para a obtenção do Grau de Engenheiro de Controle e Automação.

Orientador: Prof. Eduardo José da Silva Luz, Dr.

Coorientador: Prof. Luciana Gomes Castanheira, Dra.

Ouro Preto
Escola de Minas – UFOP
2022

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

S232m Santandrea, Alan Souza.
MLOps [manuscrito]: Introdução ao tema e estudo de caso. / Alan Souza Santandrea. - 2022.
43 f.: il.: color., tab..

Orientador: Prof. Dr. Eduardo José da Silva Luz.
Coorientadora: Profa. Dra. Luciana Gomes Castanheira.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Escola de Minas. Graduação em Engenharia de Controle e Automação .

1. Aprendizado do computador - MLOps (software). 2. Aprendizado do computador - Pipeline. 3. Administração - Melhoria Contínua. I. Castanheira, Luciana Gomes. II. Luz, Eduardo José da Silva. III. Universidade Federal de Ouro Preto. IV. Título.

CDU 681.5

Bibliotecário(a) Responsável: Maristela Sanches Lima Mesquita - CRB-1716

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
ESCOLA DE MINAS
DEPARTAMENTO DE ENGENHARIA CONTROLE E AUTOMACAO

FOLHA DE APROVAÇÃO

Alan Souza Santandrea

MLOps: Introdução ao tema e estudo de caso

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Engenharia de Controle e Automação

Aprovada em 17 de outubro de 2022.

Membros da banca

Prof. Dr. Eduardo José da Silva Luz - Orientador - Universidade Federal de Ouro Preto
Profa. Dra. Luciana Gomes Castanheira - Coorientadora - Universidade Federal de Ouro Preto
Prof. Dr. Pedro Henrique Lopes Silva - Universidade Federal de Ouro Preto
Prof. Dr. Rodrigo Cesar Pedrosa Silva - Universidade Federal de Ouro Preto

Luciana Gomes Castanheira, coorientadora do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 17/10/2022

Documento assinado eletronicamente por **Luciana Gomes Castanheira**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 17/10/2022, às 11:44, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0412724** e o código CRC **C528734D**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: 3135591533 - www.ufop.br

Este trabalho é dedicado a todos aqueles que, por medo de errarem,
nunca gozarão do prazer de acertar.

AGRADECIMENTOS

Agradeço imensamente à minha família por sempre me apoiar e me fazer seguir em frente, com um agradecimento especial aos meus tios Paulo Antônio e Guiomar de Grammont por me receberem em sua casa durante os anos iniciais da graduação e me tratarem como um filho.

Serei eternamente grato aos amigos e amigas da equipe 42 por todos os excelentes momentos que compartilhamos juntos, pelas ajudas prestadas, preocupações sanadas e boas risadas. Vocês tornaram toda a jornada mais fácil. Agradeço aos meus grandes amigos Lucas Lages, Bernardo Alves e Patty dos Reis por transformarem o apartamento 101 em algo bem maior do que apenas uma moradia compartilhada.

A todos os integrantes do TerraLAB e da equipe 12 BIS por todos os momentos que passamos juntos e desafios enfrentados.

Agradeço à treinadora Marinalva Brito e a todo o time de atletismo por me mostrarem uma nova paixão e partilharem dela comigo.

Agradeço, com muito carinho, aos professores Tiago Carneiro e Rodrigo Pedrosa por tudo que me ensinaram e pelas portas que me abriram. Sem vocês meu futuro poderia ser completamente diferente.

Finalmente, Eduardo Luz. Não consigo expor em palavras o quanto sou grato ao senhor. Agradeço por acreditar e apoiar minhas ideias mirabolantes e me guiar em todos os aspectos. Por me acalmar nos momentos difíceis e ser sempre um mentor durante toda a faculdade. Este trabalho só foi possível por conta do senhor, de todo o carinho que teve por mim e por me ensinar muito além do eu sequer imaginaria aprender. Um orientador; um professor; um grande amigo. Obrigado.

“A imaginação é mais importante que o conhecimento, porque o conhecimento é limitado, enquanto a imaginação abrange o mundo inteiro...” (Albert Einstein)

RESUMO

Cada vez mais, serviços e aplicações estão incorporando modelos de aprendizado de máquina em seu fluxo, de forma a melhorar análises ou agregar maior valor em suas respostas. Entretanto, os times e desenvolvedores encontram diversas dificuldades ao implementarem esses modelos em um ambiente de produção e se deparam com inúmeros empecilhos operacionais.

Dos esforços criados para sanar tais problemas, um que está em alta é a difusão da cultura do Machine Learning Operations (MLOps), também chamada de Operações de Aprendizado de Máquina. O presente trabalho propõem-se a fazer um introdução teórica à cultura, apresentando seus principais conceitos. O trabalho também apresenta uma revisão baseada na literatura. Indo mais além, é realizado um estudo de caso de um time na indústria, com o objetivo de demonstrar e discutir empiricamente acerca de alguns dos conceitos apresentados. O time acompanhado por essa pesquisa utiliza amplamente de modelos de aprendizado de máquina e está começando a migrar seu desenvolvimento para um cenário centrado nos princípios da cultura MLOps.

Como conclusão foi possível perceber que a vasta gama de discussões e diferentes pontos de vista acaba por tornar complicado a implementação da cultura na prática. Entretanto, o caso estudado da indícios de que uma implementação gradual com foco nas necessidades momentâneas pode ser uma saída para tornar o uso do MLOps mais fácil e eficiente.

Palavras-chaves: MLops, Pipeline de Aprendizado de Máquina, Melhoria Contínua, Operações.

ABSTRACT

Services and aplications are increasingly incorporating mahchine learning models into their workflow in order to improve analytics or add greater value to their responses. However, teams and developers come across several difficulties when deploying these models in a production enviorement and face numerous hindrances when operationalizing them.

Of the efforts created to solve such problems, one that getting attention is the dissemination of the MLOps culture, also known as Machine Learning Operations. The present work proposes to make a theoretical introduction to the culture, highlighting its main concepts. The work also presents a literature-based review. Furthermore, is accomplished a case study of a team in the industry, aiming to demonstrate and discuss empirically about some of the concepts presented. The team followed by this research makes extensive use of machine learning models and is starting to migrate its development to a scenario centered on the principles of the MLOps culture.

As a conclusion, it was possible to perceive that the wide range of discussions and different points of view ends up making the practical implementation of the culture complicated. However, the case studied shows evidences that a gradual implementation focusing on momentary needs can be a way out to make the use of MLOps easier and more effective.

Key-words: MLops, Machine Learning Pipeline, Continuous Training, Operations.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
AM	Aprendizado de máquina
AI	*Artificial Intelligence*
CD	*Continuous Deployment and Delivery*
CI	*Continuous Integration*
CT	*Continuous Training*
DS	*Data Scientist*
IA	Inteligência Artificial
ML	*Machine Learning*
MLE	*Machine Learning Engineer*
NLP	*Natural Language Processing*
QA	*Quality Assurence*
T/L	*Team Leader*
TL	*Tech Lead*
TI	Tecnologia da Informação

SUMÁRIO

1 INTRODUÇÃO

A tecnologia está evoluindo em velocidade exponencial. A cada ano que passa os avanços são tantos que fica praticamente impossível de acompanhar. Pode-se pegar exemplos que extrapolam a órbita do mundo, como os foguetes e satélites do projeto *Star Link* da Tesla ou o telescópio *James Webb*, demonstrando o avanço da engenharia e ciência.

Entretanto, todos esses projetos compartilham de uma característica em comum que não é tão perceptível e palpável. Eles geram dados. Para ser mais preciso, o *Webb* transmite no mínimo 60 gigabytes de dados por dia (INSTUTUTE, 2021). Este é apenas um dos exemplos que ajudaram a nomear os dias de hoje como a era do *Big Data*, marcada pela gigantesca produção de dados em todo o mundo. Seja dentro das casas, nas indústrias ou redes sociais estamos constantemente produzindo quantidades esmagadoras de dados. Mas para que servem? Os dados crus, como são chamados, não possuem utilidade. É preciso extrair conhecimento deles. Precisam ser organizados e depois estudados assim gerando os *insights* e é através destes que se consegue aprender. Entretanto, seria praticamente impossível para um humano analisar esta enorme quantidade de dados. Surge então outra vertente tecnológica em alta na atualidade: a inteligência Artificial - que de agora para frente será abreviada para IA. Algorítimos de IA têm como objetivo imitar nossas capacidades de raciocinar, perceber o mundo e identificar os objetos a nossa volta (TEIXEIRA, 2019).

Aprofundando no universo da IA chegamos no subconjunto chamado de aprendizado de máquina.

> "Aprendizado de Máquina é uma área de IA cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. Um sistema de aprendizado é um programa de computador que toma decisões baseado em experiências acumuladas através da solução bem sucedida de problemas anteriores."(MONARD; BARANAUSKAS, 2003, p.39).

É nesta vertente que o presente trabalho pretende atuar. Modelos de aprendizado de máquina (AM) são treinados em cima de dados anteriores com o objetivo de inferir corretamente os novos. Supondo um modelo que tem como objetivo classificar imagens entre as classes gato e cachorro, o modelo será treinado em diferentes imagens dessas classes e quando for exposto a uma imagem nunca antes vista, irá rotulá-la como sendo de uma das duas: gato ou cachorro. O exemplo anterior é um problema típico de classificação, mas existem muitos outros como: predição, agrupamento, recomendação, etc. Em suma, a área de AM vem atraindo cada vez mais a atenção de pesquisadores e industrias que veem nela uma saída para diversos problemas atuais. Com esse frenesi de utilização, diversas ferramentas, métodos e modelos estão sendo disponibilizados para os mais diversos problemas.

Alguns dos mais recentes estudos estão ligados a áreas como: detecção de objetos (GHIASI et al., 2020), (DAGLI; SHAIKH, 2021), (ZHANG et al., 2022), (ZHENG et al., 2022); classificação de imagens (YU et al., 2022), (TOUVRON et al., 2021), (KABIR et al., 2020), (FORET et al., 2020); processamento de linguagem natural (HENDRYCKS et al., 2021), (JIANG et al., 2020), (RAFFEL et al., 2019), (SHOEYBI et al., 2019); para problemas médicos (BENčEVIć et al., 2021), (ZHOU et al., 2020), (AKEN et al., 2021), (ALPERSTEIN; CHERKASOV; ROLFE, 2019), (ZHANG et al., 2022).

O parágrafo anterior expõe uma ínfima quantidade de modelos, métodos e ferramentas que objetivam resolver problemas com AM disponibilizados na literatura. Surgem agora os questionamentos: Como uma empresa consegue utilizar de forma constante e efetiva tudo a que está sendo exposta? Quais as ferramentas certas, técnicas, dados, modelos. Como atualizar para um modelo melhor? Quando atualizar? E finalmente, como manter tudo isso organizado para que seja possível rastrear, reproduzir e compartilhar entre equipes? Como operacionalizar o aprendizado de máquina?

Estas mesmas perguntas foram feitas para softwares tradicionais há não muito tempo atrás. Dessas indagações surgiu o DevOps, que possui como premissa diminuir as dificuldades no desenvolvimento de softwares e é mais do que apenas uma metodologia - como é o caso do método em cascata (ROYCE, 1987) ou o manifesto ágil (BECK et al., 2001) - é uma cultura e um paradigma que engloba problemas técnicos e sociais de uma organização comprometida a desenvolver softwares (KREUZBERGER; KÜHL; HIRSCHL, 2022). As experiencias e técnicas adquiridas por especialistas e desenvolvedores durante os últimos anos com o DevOps são agora utilizadas para o desenvolvimento, automação e operacionalização de modelos de AM. Com o objetivo de responder a todas as perguntas do parágrafo anterior surge o CRISP-ML(Q), um modelo de processo que visa melhorar o modo como modelos são criados e colocados em produção e o MLOps (*Machine Learning Operations*, traduzido livremente para Operações de Aprendizado de Máquina), uma cultura embasada nos métodos ágeis, no DevOPS, no CRISP-ML(Q) e em metodologias inovadoras que está aos poucos ganhando espaço na industria e na academia, ao passo que modelos de AM se tornam cada vez mais comuns nas aplicações.

1.1 Objetivos Gerais

Diante do exposto, este trabalho propõem fazer uma revisão em cima do tema de MLOps, introduzindo o que está sendo discutido na área, com finalidade de tornar todo o conhecimento espalhado útil para a indústria e outros. Também é proposto o estudo de um caso (VENTURA, 2007) real na indústria. Que fique claro que a ideia não é criar uma metodologia de base e sim apresentar o que está sendo discutido em cima do tema.

1.2 Objetivos Específicos

Os objetivos mais específicos são:

- Revisar brevemente o conceito de Aprendizagem de Máquina.
- Revisar brevemente os conceitos do DevOps.
- Revisar o conceito de CRISP - ML(Q).
- Revisar o conceito de MLOps do ponto de vista de produto e desenvolvimento.
- Expor e analisar um caso real na indústria.

2 REVISÃO DA LITERATURA

2.1 Aprendizado de Máquina

Em 1959, Arthur Samuel deu a seguinte definição, considerada a primeira do mundo: "Aprendizado de máquina é o campo de estudo que garante aos computadores a habilidade de aprender sem que seja explicitamente programado."Em 1997, 38 anos mais tarde, Tom Mitchell expõe de uma maneira mais orientada à engenharia o seguinte: "Um programa de computador é dito aprender de uma experiência E em relação a uma tarefa T e alguma medida de desempenho P, se seu desempenho em T, medido por P, melhora com a experiencia E."(GÉRON, 2019, pg.2, tradução nossa).

Um algoritmo de aprendizado de máquina também conhecido como ML (do inglês *Machine Learning*), tem como objetivo realizar uma tarefa sem que seja explicitamente programado para tal, como é o caso de sistemas tradicional com vários blocos condicionais. Ao invés disso, o algoritmo recebe como entrada um conjunto de dados, passa por várias iterações de treino em cima deles a fim de otimizar seu desempenho na tarefa e acaba por aprender a realizar essa tarefa, concluindo assim seu objetivo inicial. A Figura 1 mostra de forma simplificada o que foi descrito.

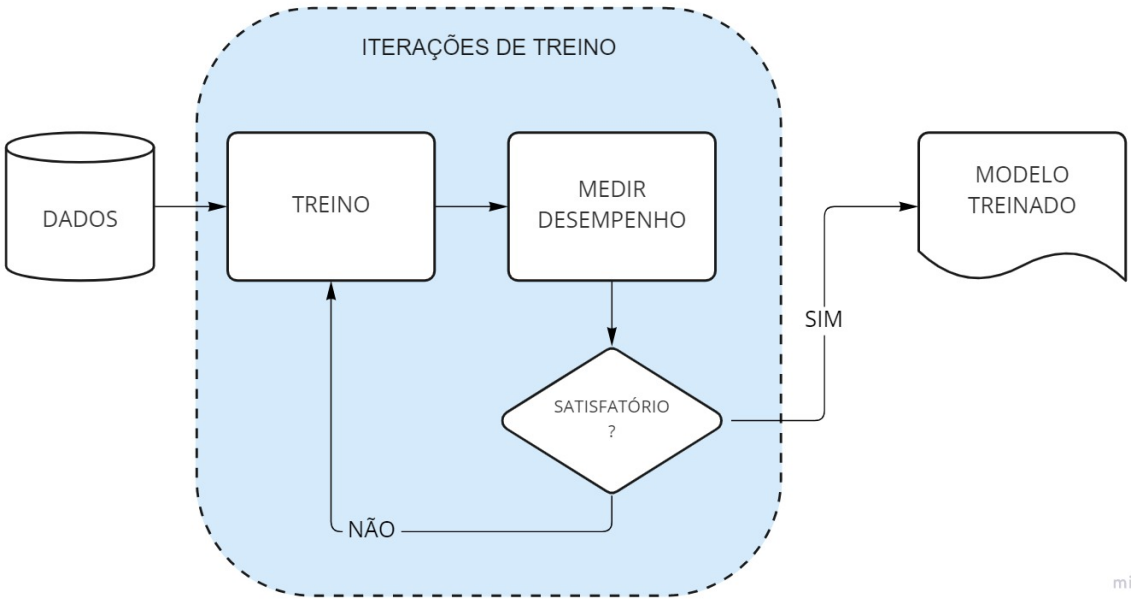

Figura 1 – Diagrama explicativo da concepção de um algorítimo de aprendizagem de máquina. Fonte: De autoria própria.

O que comumente gera confusão, é o fato de que o ML é um ramo de estudo dentro da IA e não um sinônimo dela. Portanto, todo projeto de ML é um projeto de IA, mas nem todo projeto de IA é alimentado por um modelo de ML. 2

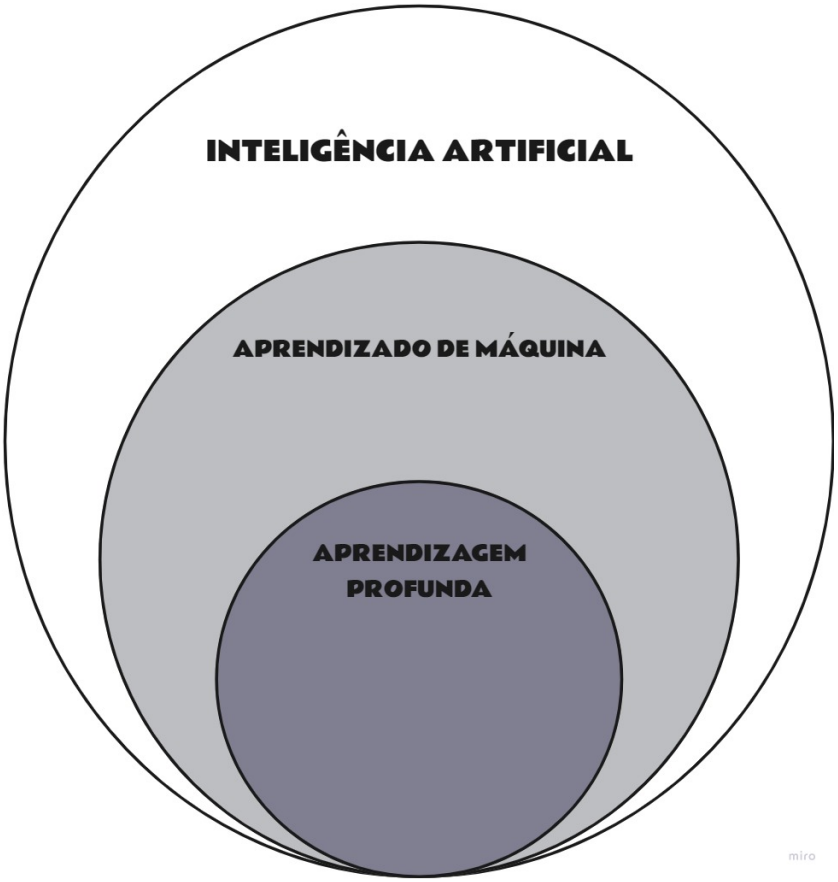

Figura 2 – Universo de IA e sua sub divisões. O círculo mais externo representa o universo de IA, o do meio é o subconjunto de modelos de Aprendizado de Máquina e o mais interno de todos são os modelos de aprendizado profundo. Fonte: De autoria própria

Os sistemas de Aprendizado de Máquina podem ser divididos em diferentes tipos por diferentes critérios. Quanto ao tipo de supervisão são divididos em: supervisionados, não supervisionados, semi-supervisionados e de aprendizado por reforço. Além disso podem ser online ou offline. Entretanto, é possível encontrar sistemas com 2 ou mais tipos interligados (IPPOLITO; FERGUSON; JENSON, 2021),(ARORA et al., 2018).

2.2 DevOps

A criação de um projeto de software pode ser, simplificadamente, dividida entre dois grandes times: (I) desenvolvimento (Dev) e (II) operação (Ops) (PERERA; SILVA; PERERA, 2017). O primeiro é responsável por pensar o como, fazer as experimentações e escrever o código da aplicação; enquanto o segundo tem como função testar, colocar em produção e coletar feedbacks da aplicação para o primeiro. Durante muito tempo era comum que estes times atuassem de forma separada. Os desenvolvedores criavam o artefato que seria passado para o time de operação e, enquanto este trabalha para colocar o artefato em produção e começar a coletar as avaliações, aquele ficaria ocioso. O time de desenvolvimento algumas das vezes até ocuparia esse tempo 'livre' iniciando outro projeto, mas assim que o time de operação retornasse

com feedbacks pedindo funcionalidades ou aprimoramentos no artefato antigo, fatalmente seria necessário parar o novo projeto.

Consequentemente, o fluxo de desenvolvimento descrito acima era extremamente defeituoso e difícil de manter e escalar, além de ser demorado do ponto de vista de entregas ao cliente. Imagine a situação onde se tem apenas um time de operação para vários times de desenvolvimento. Ou o tempo ocioso do desenvolvimento aumenta ou o time de operação fica extremamente sobrecarregado. Foi então que, em concordância com a disseminação das metodologias ágeis (BECK et al., 2001), surgiu o DevOps. Uma cultura que tem como principal finalidade identificar e eliminar as lacunas entre o time de desenvolvimento e o time de operação e garantir um melhor fluxo de evolução de um projeto de software para entregar valor mais rapidamente (PERERA; SILVA; PERERA, 2017). A Figura 3 ilustra o símbolo do DevOps, inspirado no símbolo do infinito, onde a ideia é mostrar a conexão entre os dois times com um vínculo contínuo e ininterrupto.

O DevOps tem como um de seus mais importantes princípios a velocidade com a qual o produto chega nas mão do cliente. A ideia é entregar rapidamente um MVP (do inglês *Minimum Valuable Product* traduzido para Mínimo Produto Viável) e fazer vários ciclos de melhoria nesse produto. Para isso, a automação é extremamente importante e é garantida, em grande parte, pela prática da integração e entrega contínua (CI/CD), permitindo ciclos de entregas mais rápidos, frequentes e confiáveis. Além disso o DevOps é idealizado para ter garantia de qualidade (QA - Quality Assurance), testes contínuos, monitoramento contínuo, *logging* e laços de retorno (*feedback loops*) (KREUZBERGER; KÜHL; HIRSCHL, 2022).

Que fique claro que o DevOps é um processo relativamente novo na industria e ainda não é completamente padronizado. Portanto, quais métodos ou ferramentas a empresa utiliza, são extremamente dependentes de cada caso e da escolha de quem esta implementando. Se os métodos adotados pela empresa são suficientemente rápidos e eficientes para o que ela se dispõem, então não tem por que mudá-los (MLOPS-SIG, 2022).

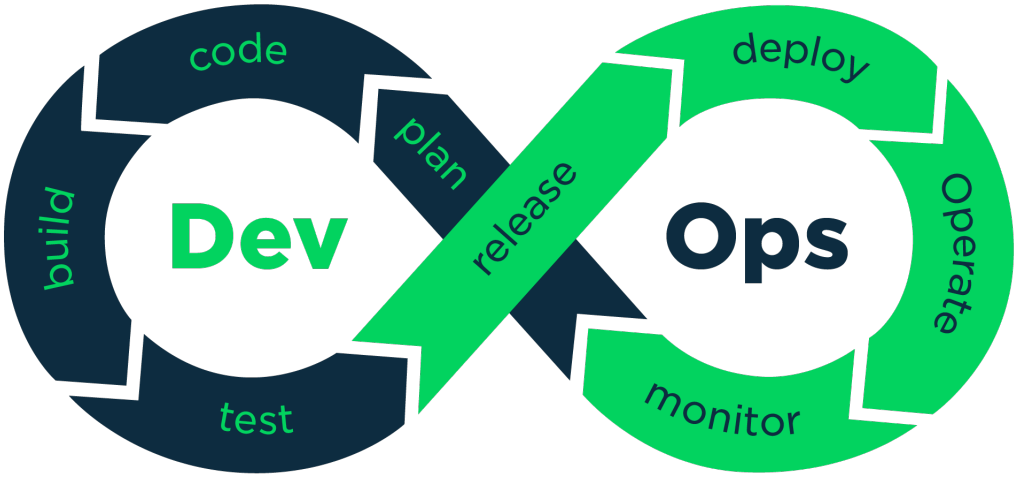

Figura 3 – Diagrama de vetores que ilustra o modo de pensar DevOps. Fonte: https://blog.mandic.com.br/artigos/como-o-DevOps-afeta-o-trabalho-do-cio/

2.3 CRISP-ML(Q)

O CRISP-ML(Q) (*CRossIndustry Standard Process model for the development of machine learning applications with Quality assurance methodology*) é um modelo de processo definido por Studer et al. (2021) que, baseado no CRISP-DM (WIRTH; HIPP, 2000), foca principalmente nas tarefas técnicas, necessárias para produzir evidências de que todos os passos do processo de desenvolvimento têm qualidade o suficiente para justificar a adoção no processo de negócio.

O artigo define seis fases para o desenvolvimento e como primeira contribuição, introduz uma metodologia de garantia de qualidade, ilustrada na Figura 4, em toda tarefa ou fase do modelo de processo. Como uma segunda contribuição, o CRISP-ML(Q) define uma fase de monitoramento e manutenção para prevenir riscos de degradação do modelo de ML em ambientes de constante mudança - etapa inexistente no CRISP-DM. Portanto, todos os itens que compõem o modelo de processo são, em ordem: (i) *Business & Data Understanding; (ii) Data Preparation; (iii) Modeling; (iv) Evaluation; (v) Deployment; (vi) Monitoring & Maintenance*, traduzidas livremente para: Entendimento do Negócio e dos Dados; Preparação dos Dados; Modelagem; Avaliação; Implementação; Monitoramento e Manutenção.

O artigo então, aprofunda em cada uma das fases e desenvolve tarefas a serem cumpridas em cada uma, juntamente com critérios de aceite e possíveis riscos. Como exemplos podem ser citados: discutir critérios de sucesso e viabilidade para o item (i); selecionar dados e engenharia de atributos na fase de preparação dos dados do item (ii); realizar pesquisas na literatura ou problemas similares, definir medidas de qualidade e treinar o modelo no item (iii); validar o desempenho e comparar resultados com um critério de sucesso no item (iv); estratégias de implementação e minimizar riscos de erros inesperados no item (v); monitorar e atualizar no

item (vi).

O modelo de processo é extremamente útil e engloba várias etapas do desenvolvimento. Por isso, se seguido à risca, garante que o projeto de ML esteja em um padrão industrial, diminuindo consideravelmente suas chances de abandono por não se comportar de forma satisfatória ao sair do ambiente controlado de experimentação e ir para o ambiente de produção.

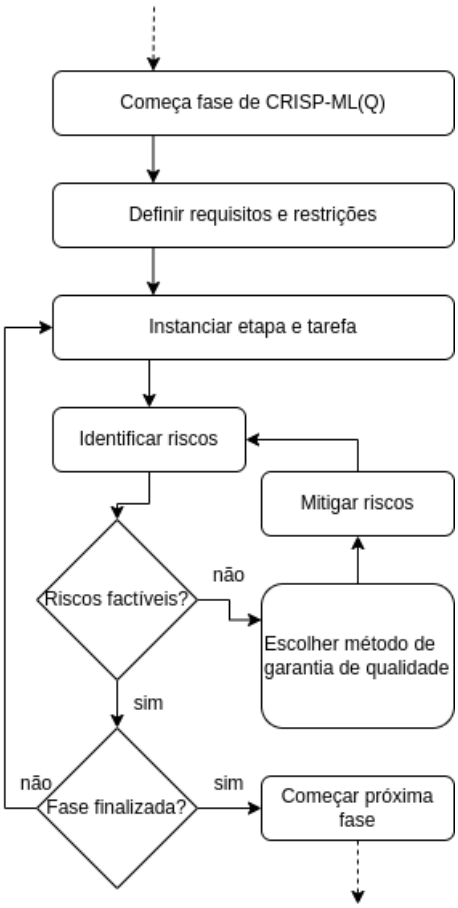

Figura 4 – Imagem da abordagem do CRISP-ML(Q) para garantia de qualidade. O diagrama mostra os passos para uma tarefa da fase de desenvolvimento. Fonte: Adaptado de (STUDER et al., 2021)

3 MLOPS

3.1 Introdução ao MLOps

Baseando-se no DevOps, o objetivo do MLOps pode ser considerado o mesmo: diminuir as lacunas entre desenvolvimento e operação. Entretanto, aqui se discute sobre o desenvolvimento e operacionalização de softwares alimentados por modelos de aprendizado de máquina. Esses tipos de softwares tendem a ser muito mais difíceis de manter, testar e de garantir a qualidade. Além disso sofrem de forma mais intensa com os débitos técnicos, como foi discutido por Sculley et al. (2015).

Entretanto, vale ressaltar que existem poucos trabalhos publicados na área e que a maior parte do conhecimento sobre o assunto advém de comunidades e blogs na internet, entusiastas do assunto e startups. Dito isso, MLOps-SIG (2022) define MLOps como "A extensão da metodologia DevOps para incluir disciplinas da Aprendizado de Maquina e Ciência de Dados como *first-class citizens*[1] dentro do DevOps."

Portanto, não existe ainda um padrão ou consenso sobre o que é *Machine Learning Operations* e quais os passos exatos a serem seguidos. A literatura atual tende a discutir sobre problemas mais técnicos do pipeline de desenvolvimento, as etapas que o constituem e sobre as ferramentas que aparecem aos montes propondo "facilitar"algumas dessas etapas. (SYMEONIDIS et al., 2022), (KREUZBERGER; KÜHL; HIRSCHL, 2022), (HEWAGE; MEEDENIYA, 2022), (TESTI et al., 2022).

Existem trabalhos que, por falta de material teórico sobre o assunto, realizaram entrevistas com profissionais da área, de forma a estarem mais alinhados com o dia a dia das empresas e como se dá a execução dentro delas (KREUZBERGER; KÜHL; HIRSCHL, 2022),(SHANKAR et al., 2022).

Doravante, o presente trabalho, baseando-se na literatura e material disponíveis, define MLOps como: O conjunto de ações e operações minimamente indispensáveis e necessárias para desenvolver, testar, colocar em produção e monitorar um modelo de aprendizagem de máquina em produção, com velocidade qualidade e com o menor atrito possível entre os times de desenvolvimento e operacionalização.

Indo além, neste trabalho será apresentado o MLOps sob dois pontos de vista - produto e desenvolvimento.

[1] Traduzido livremente como Cidadão de Primeira Classe, em uma determinada linguagem de programação é uma entidade que suporta todas as operações geralmente disponíveis para outras entidades. Disponível em https://pt.wikipedia.org/wiki/Cidad~ao_de_primeira_classe

3.2 MLOps do ponto de vista de produto

Esse ponto de vista vai desde o concebimento da ideia até o monitoramento do produto final, porém sempre objetivando facilitar a produtização. A Figura 5 mostra um diagrama das etapas.

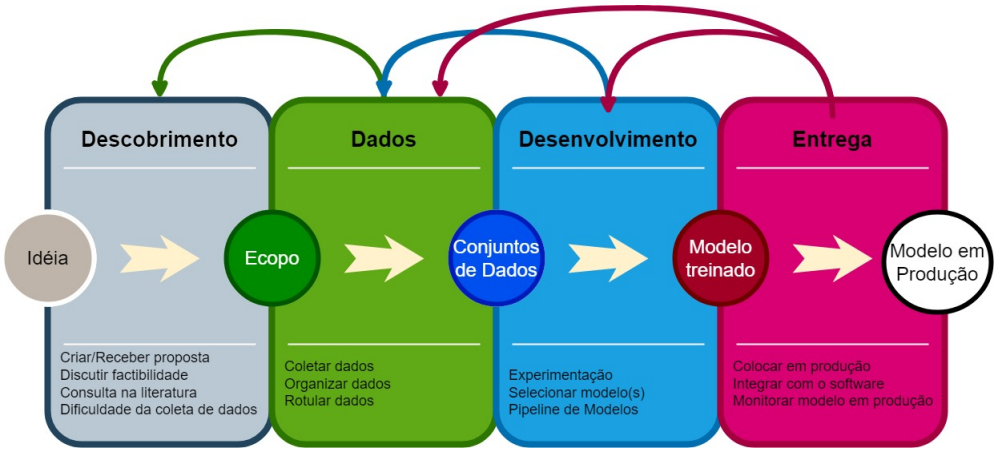

Figura 5 – Ciclo de vida do aprendizado de máquina do ponto de vista de produto. Fonte: De autoria própria

A primeira etapa é a de descobrimento. Ela tem como entrada uma ideia, que pode ser interna da empresa ou um pedido de cliente. Aqui cabe discutir se o projeto é factível ou não, qual a dificuldade em se coletar os dados, consultar a literatura procurando os modelos que são estado da arte a fim de descobrir se já existem trabalhos em cima do que está sendo idealizado e qual a qualidade dos resultados obtidos. Esta etapa é normalmente exercida por gerentes de produtos, líderes de times e líderes técnicos, mas depende muito de como a empresa funciona internamente. O artefato desta etapa é o escopo do projeto pronto contendo as diretrizes a serem seguidas e metas a serem alcançadas (tanto métricas de produto como também métricas de negócio). É possível já prever como os dados devem sair, quais modelos serão testados e qual o mínimo que se espera do modelo ou que o cliente concordou como aceitável, entre outros fatores.

Para facilitar, pegue como exemplo a seguinte situação: um cliente, que é uma empresa que permite investir em cripto-ativos através de seu site, pediu um modelo para previsão da bolsa de valores. Tem-se da literatura que esta tarefa é extremamente difícil, senão impossível (ZHANG; XU; XUE, 2017), (HASSAN; NATH; KIRLEY, 2007). O time então discute com o cliente qual o objetivo do projeto. O cliente então, diz que não precisa de previsões perfeitas pois ele pretende usa-las em um sistema de aviso para prevenir que investidores inexperientes percam dinheiro. No caso, se o investidor executa a ação de comprar um ativo, mas o modelo previu que este está em um tendência de baixa, o sistema avisa o usuário antes de confirmar a transação. No exemplo anterior uma métrica importante é a precisão. O modelo não pode deixar de avisar uma tendência de baixa pois isso causaria perda de dinheiro e abandono da plataforma, portanto é

plausível que seja acordado uma precisão de no mínimo 75%. Então, após pesquisar na literatura, o time constata que tal métrica é alcançável e o projeto se torna factível, podendo seguir para a próxima etapa na esteira de ações com alguns limites e definições previstas no escopo.

A segunda etapa é a dos dados. Aqui o time de dados ou responsáveis irão coletar bases de dados para treinar o modelo em questão. Também ocorre a limpeza, manipulação, *feature engineering* (engenharia de atributos, tradução livre) e colocação de rótulos para modelos de aprendizagem supervisionada. O artefato desta etapa são os conjuntos de dados, normalmente distribuídos em treino, validação e teste.

Seguindo o mesmo exemplo do cliente - empresa de investimentos- o time coletaria os dados de cripto-moedas no tempo, a fim de criar os conjuntos de dados para treino de um modelo de previsão. No caso do exemplo, o modelo deve prever a tendência. Um jeito seria coletar valores do preço em intervalos ou coletar janelas de preços em intervalos fixos e associar a cada intervalo um rótulo de tendência da próxima janela. A API do cliente não irá entregar os dados dessa maneira. Ela irá entregar um conjunto de dados históricos e cabe ao time organizá-los da maneira escolhida. Tamanhos da janela, quantidade de dados, onde buscar esses dados, tudo isso são tarefas cabíveis dentro da etapa.

A terceira e penúltima etapa é a de desenvolvimento. O desenvolvimento aqui refere-se exclusivamente à criação do modelo de aprendizado de máquina não tendo nenhuma ligação com o desenvolvimento da aplicação que irá consumir esse modelo. No escopo do projeto já deve constar algumas ideias de modelos a serem avaliados e é aqui que estes e outros modelos serão treinados, avaliados, testados e validados. Esta etapa é extremamente experimental e se difere fortemente do desenvolvimento de um software comum. Raramente o primeiro modelo testado é suficientemente bom, portanto as vezes são necessárias várias iterações. É nesta etapa que se concentra a maior parte do trabalho dos cientistas de dados e engenheiros de aprendizado de máquina. O artefato desta é um modelo treinado, respeitando todos os limites e requisitos descritos no escopo.

A última etapa é a da entrega. Ao final desta etapa ter-se-á um modelo em produção pronto para ser consumido, onde novos desafios aparecerão, muitos deles análogos aos desafios de aplicações tradicionais e por conta disto, aqui é de grande valia utilizar da cultura do DevOps. Mas, surgirão também, problemas específicos da operacionalização desses modelos em produção e com eles a necessidade do MLOps (PALEYES; URMA; LAWRENCE, 2022). Esses problemas são discutidos de forma mais técnica na próxima seção.

No caso do projeto fictício do exemplo, uma entrega poderia ser feita tanto do modelo como um pacote, mas nesse caso a responsabilidade de manter o modelo em produção é do cliente, como poderia ser feita através de uma interface de programação de aplicações (API RESTful) onde o cliente faz chamadas em uma aplicação e recebe como saída a resposta do modelo. O formato dessa entrega também já deve ter sido acordado na primeira etapa do projeto, bem como a infraestrutura que serve o modelo de forma a atender requisitos de taxa de

transferência (*throughput*) e latência (*latency*).

Entretanto, observando a Figura 5 é possível ver que durante a execução dos passos, o fluxo de produto pode voltar a um passo anterior, representado pelas setas que conectam etapas posteriores às anteriores.

Durante a etapa de Dados é possível voltar para a etapa de descobrimento. Digamos que durante a coleta de dados, por algum motivo, não foi possível coletar quantidade suficiente ou de boa qualidade de um conjunto de dados ou um projeto que na literatura se provou possível para dados de uma determinada região, mas que se torna extremamente difícil ou custoso coletar estes mesmos dados para a região desejada.

Durante a etapa de Desenvolvimento é possível voltar para a etapa de Dados. Às vezes somente para melhorar a qualidade dos dados coletados, ou então por ser necessário um volume maior de dados para treinar um modelo específico, ou um dos rótulos tem poucos exemplos no conjunto de dados.

E da última etapa, a Entrega, é possível voltar para as etapas de Desenvolvimento caso um modelo não esteja performando da maneira desejada em produção ou até mesmo diretamente para a etapa de Dados, sendo que, neste caso, uma possível causa é que um novo rótulo apareceu nos dados reais que não estavam presentes nos conjuntos coletado durante a etapa de Dados ou qualquer outro motivo que torna os dados inutilizáveis.

3.3 MLOps do ponto de vista de desenvolvimento

Do ponto de vista de desenvolvimento cabe falar de aspectos mais técnicos. Aqui é desenvolvido e idealizado algumas boas práticas em quase todas as etapas da Figura 5, com exceção da primeira, pois nela não existe desenvolvimento de um código.

O MLOps do ponto de vista mais técnico é comumente referido como *machine learning pipeline*, traduzido livremente para pipeline de aprendizado de máquina. Entretanto, a comunidade acadêmica e a literatura ainda se diferem fortemente quanto a forma e os componentes que compõem esse fluxo. Existem pipelines mais simples e modestos como o definido por Symeonidis et al. (2022) e ilustrado na Figura 6. Em sua obra, Hewage e Meedeniya (2022) vão um pouco além e definem juntamente com as etapas, os cargos daqueles responsáveis por exercê-las, onde já se torna possível perceber, através da Figura 7, a forte influência de componentes do DevOps e a utilização de QA como definida por Studer et al. (2021). Finalmente Kreuzberger, Kühl e Hirschl (2022), em sua proposta de pipeline, idealizam o que pode ser considerado o mais completo até a escrita do presente trabalho. Na Figura 8 é possível ver que o autor atribui a cada componente o respectivo papel ou cargo da pessoa que irá trabalhar nele, bem como desenvolve por completo cada etapa e componente. A partir da análise e leitura do artigo, pode-se argumentar que a complexidade que uma arquitetura baseada no MLOps chega a alcançar tanto em termos de componentes quanto de cargos colaborando em conjunto, é extremamente alta.

Além disso, ficam visíveis as diferenças para com o DevOps, como versionamento de modelos, dados e pipelines internos; o aparecimento de novos componentes como *Feature Stores*, *Model Registry* e principalmente a adição do retreino contínuo, que aparece somado ao CI/CD e recebe a sigla CT do inglês *Continuous Training*.

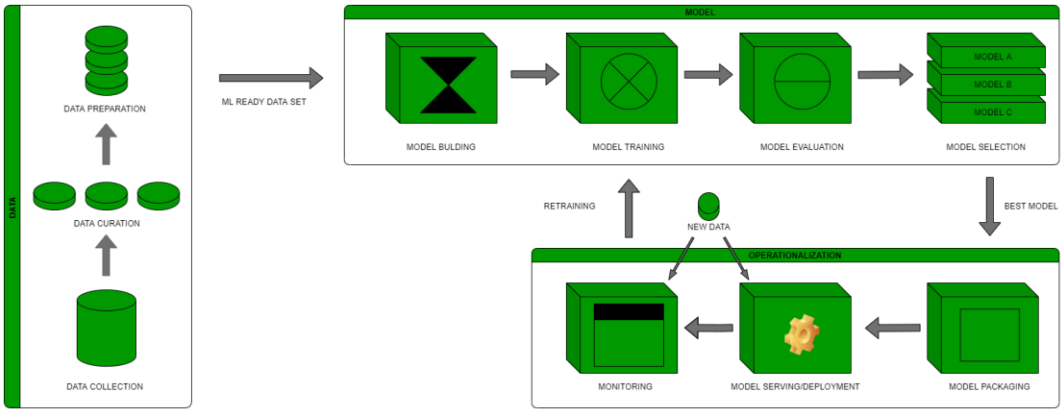

Figura 6 – Pipeline de MLOps simples, com alguns poucos componentes definidos. Fonte: (SYMEONIDIS et al., 2022)

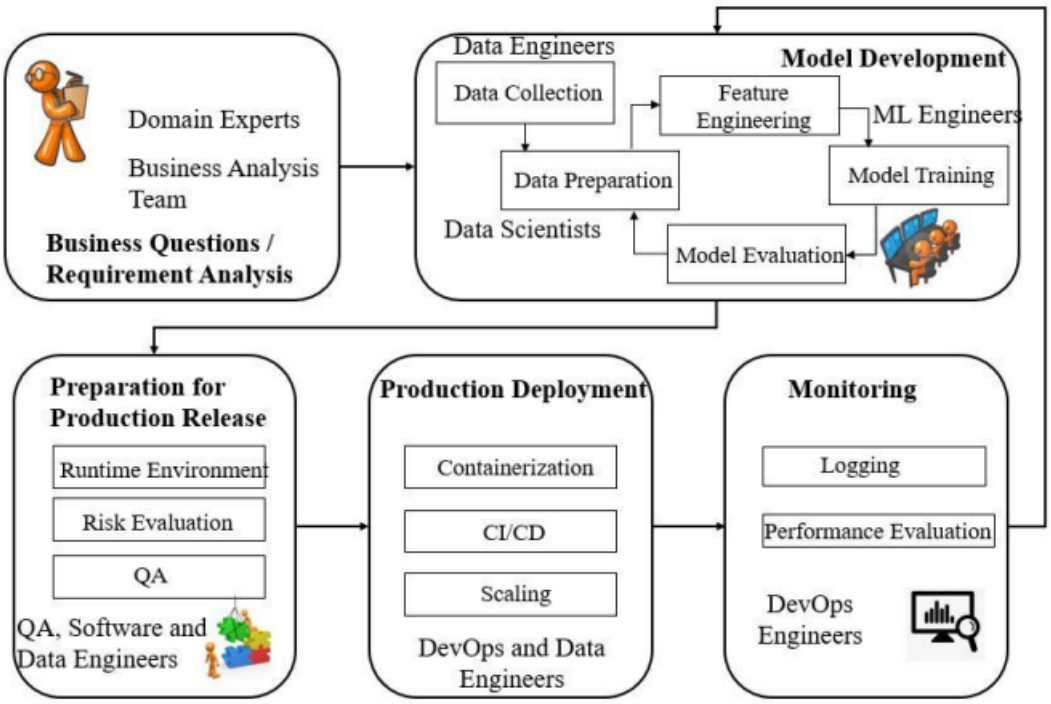

Figura 7 – Pipeline de MLOps com complexidade moderada, com número de componentes moderados e cargos associados a cada fase. Fonte: (HEWAGE; MEEDENIYA, 2022)

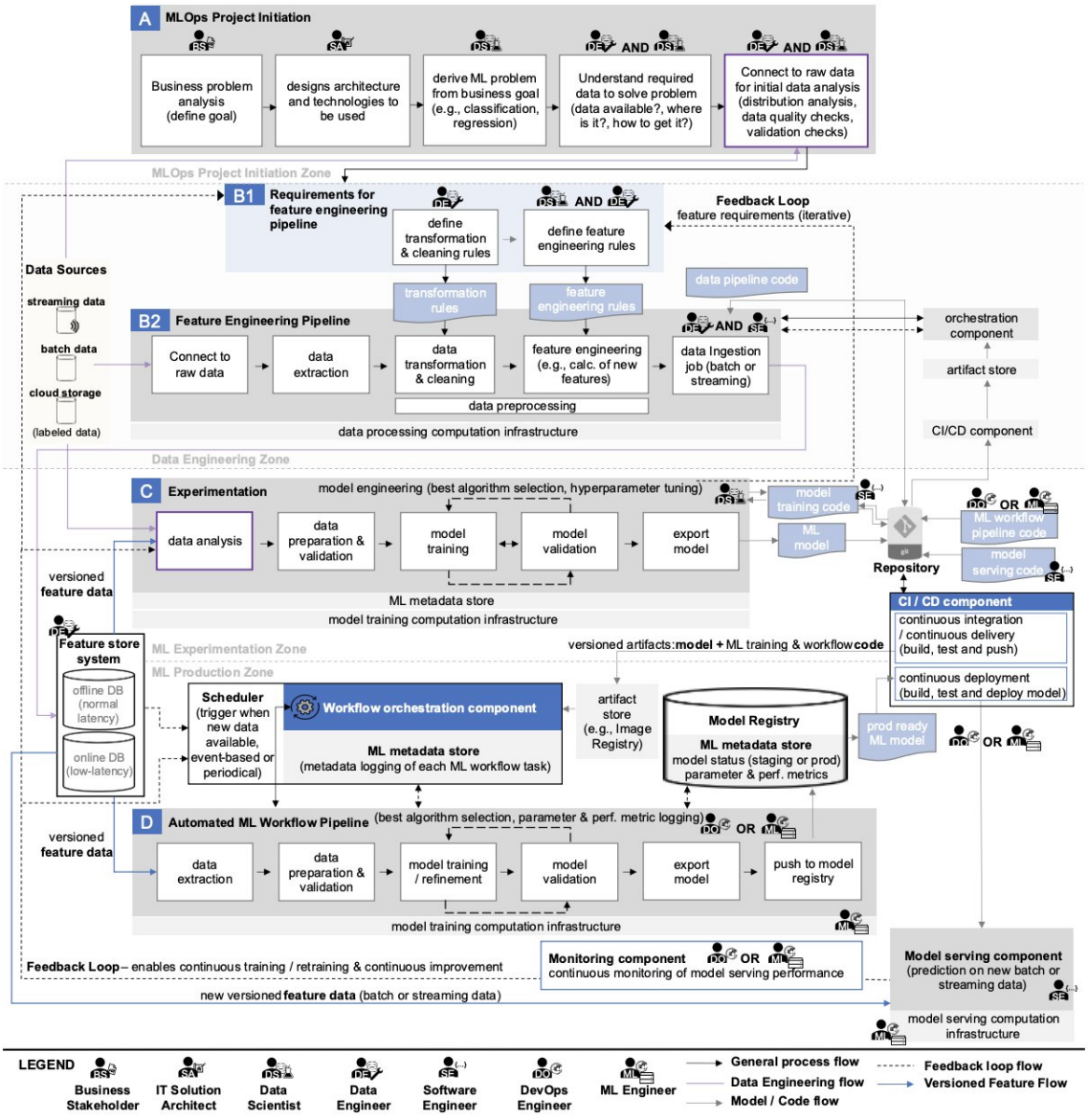

Figure 4. End-to-end MLOps architecture and workflow with functional components and roles

Figura 8 – Pipeline de MLOps complexo e bem definido, com número elevado de componentes e cargos associados a um deles. Fonte: (KREUZBERGER; KÜHL; HIRSCHL, 2022)

3.4 Princípios do MLOps

Kreuzberger, Kühl e Hirschl (2022) em sua obra definem 9 princípios para o MLOps. Nesse caso, a palavra princípio deve ser entendida como sendo um guia de como realizar as ações, algo como "boas práticas" para a indústria. Os parágrafos a seguir apresentam os 9 princípios definidos, seguidos por uma explicação do que o autor do presente trabalho, entende por cada um deles.

Automação de CI/CD. O termo já foi definido anteriormente e significa integração,

entrega e implementação contínuas. Garante um retorno rápido aos desenvolvedores acerca do sucesso ou falha de alguns passos na esteira de desenvolvimento, aumentando assim a produtividade geral.

Orquestração do fluxo de trabalho. Coordena as tarefas do pipeline de ML de acordo com grafos acíclicos dirigidos (DAGs - *Directed Acyclic Graph*). Na Figura 8 é possível identificar essa DAG pela letra 'D' e o componente de orquestração logo acima junto do agendador, que é o responsável por ativar o componente de acordo com alguma regra definida. O pipeline interno definido em 'D' não possui realimentação, por isso é acíclico [2]. Outra característica importante, é que a ordem de execução das tarefas leva em conta as relações e dependências entre elas.

Reprodutibilidade. Direto ao ponto, refere-se à capacidade de reproduzir um experimento de ML e obter exatamente os mesmos resultados todas as vezes. Garantir esse princípio permite que vários desenvolvedores consigam trabalhar em um mesmo projeto de forma eficaz.

Versionamento. No DevOps é muito comum o versionamento de códigos, mas aqui, esse princípio se estende para modelos, dados e até pipelines inteiros. Isso permite não apenas a reprodutibilidade acima citada como também a rastreabilidade.

Colaboração. Também comum ao DevOps, a colaboração garante a possibilidade de se trabalhar em conjunto nos modelos, dados e códigos. Além disso, uma cultura colaborativa tende a diminuir os silos organizacionais entre diferentes papeis.

Treino e avaliação contínuos de ML. Muitas vezes referido como CT, entra como uma extensão ao CI/CD. Significa treinar um modelo periodicamente a fim de melhorá-lo ou ajustá-lo aos novos dados. O CT só se torna possível com o suporte do componente de monitoramento, os laços de realimentação (*feedback loops*) e o máximo de automação que se conseguir no pipeline. A avaliação também ocorre continuamente para metrificar a mudança na qualidade do modelo.

***Tracking/logging* de metadados de ML**. *Tracking* pode ser traduzido livremente como rastrear e *logging* é popularmente conhecido na computação como a tarefa de guardar um registro com informações relevantes acerca de um evento. Nesse caso, é preciso rastrear e manter registros de metadados dos experimentos que culminaram em um modelo, tornando possível a total rastreabilidade dos experimentos.

Monitoramento contínuo. Consiste em ter uma forma de periodicamente averiguar dados, modelos, códigos e recursos ligados à infraestrutura e desempenho do serviço que serve os modelos (como por exemplo acurácia), com a finalidade de identificar erros ou mudanças fora do comportamento esperado e que influenciam a qualidade do produto. O monitoramento contínuo garante o CT e é exatamente quem irá acioná-lo assim que uma necessidade de retreino

[2] Apesar de na imagem a DAG apresentar uma realimentação entre os componentes de treino e validação (*Model training/ refinement e model validation*), isso só ocorre para ilustrar que é possível fazer várias experimentações a fim de determinar um melhor modelo. Colocar o fluxo com uma DAG facilita muito a explicação e implementação, mas por ser um campo de estudo novo, grande parte dessas definições podem, muito provavelmente, não estarem corretas e ainda passarão por diversas modificações que podem simplificar ou complicar ainda mais o fluxo.

for identificada.

Laços de realimentação. São necessários para integrar as necessidades de melhorias nos códigos e modelos, A Figura 8 mostra um exemplo onde a tarefa de avaliação do modelo ativa um laço de realimentação para a tarefa de treinamento a fim de melhorar o modelo. Outro laço de realimentação muito importante é o que vai do componente de monitoramento para o agendador citado no parágrafo **Orquestração do fluxo de trabalho**.

3.5 Níveis de maturidade

Dependendo de alguns critérios como nível de automação do fluxo de MLOps ou a presença ou não de certos componentes, um projeto pode ser colocado em diferentes níveis nomeados pela comunidade como níveis de maturidade. Apesar de não haver consenso acerca de uma matriz de maturidade universal, as duas mais aceitas são as criadas pela Google e pela Microsoft. No modelo da Google (GOOGLE, 2020) existem três níveis de maturidade e sua estrutura é apresentada na Figura 9.

Nível 0 - processo manual. É o nível comum para empresas que estão começando a aplicar ML aos casos de uso. Também chamado de nível básico, onde o processo de criação e operacionalização de um modelo é feito manualmente. Este nível é adequado para times que precisam colocar novos modelos em produção apenas uma ou duas vezes ao ano e não justificaria gastar recursos para elevar o nível de automação.

Nível 1 - automatização do pipeline de ML. Elevar a maturidade para o primeiro nível significa ser capaz de realizar o treinamento contínuo do modelo através da automatização do pipeline de ML. É um salto gigantesco se comparado ao nível anterior e torna a produção de modelos mais rápida e robusta. Com esse nível os times conseguem colocar novos modelos em produção em um curto espaço de tempo e com qualidade garantida.

Nível 2 - automatização do pipeline de CI/CD. O mundo perfeito. Todas as etapas, inclusive os pipelines, passam pelo processo de CI/CD automatizado. Os cientistas de dados ou outros cargos conseguem experimentar novas ideias de forma rápida e o sistema mantém seu modelo atualizado automaticamente através do treinamento contínuo. Esse nível é extremamente difícil de ser alcançado e talvez para muitas empresas e times nem valha o esforço de implementá-lo.

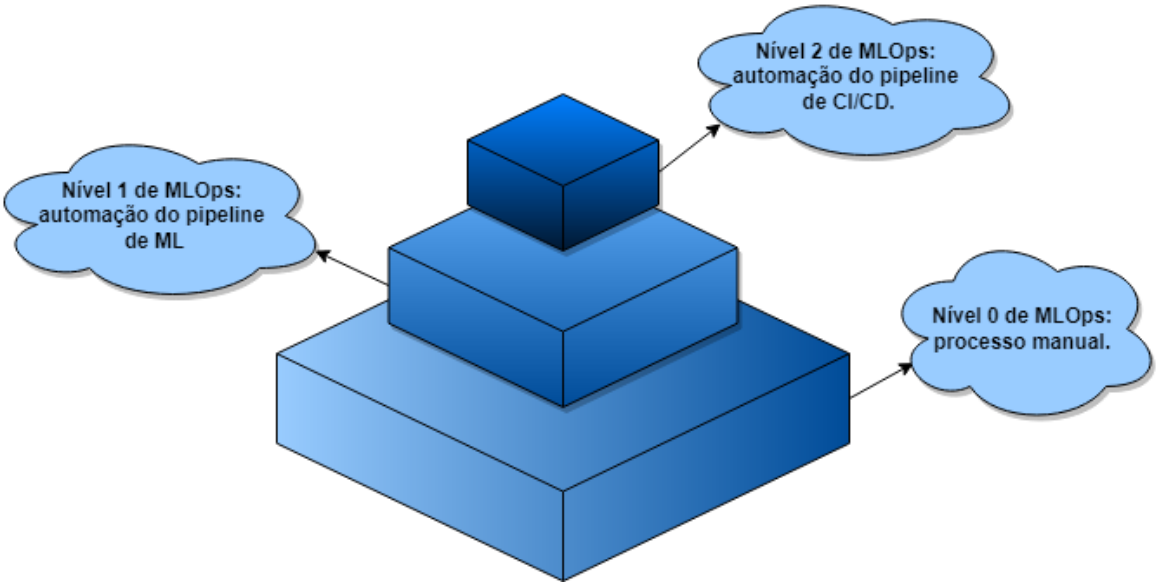

Figura 9 – Níveis de Maturidade da Google. Fonte: De autoria própria

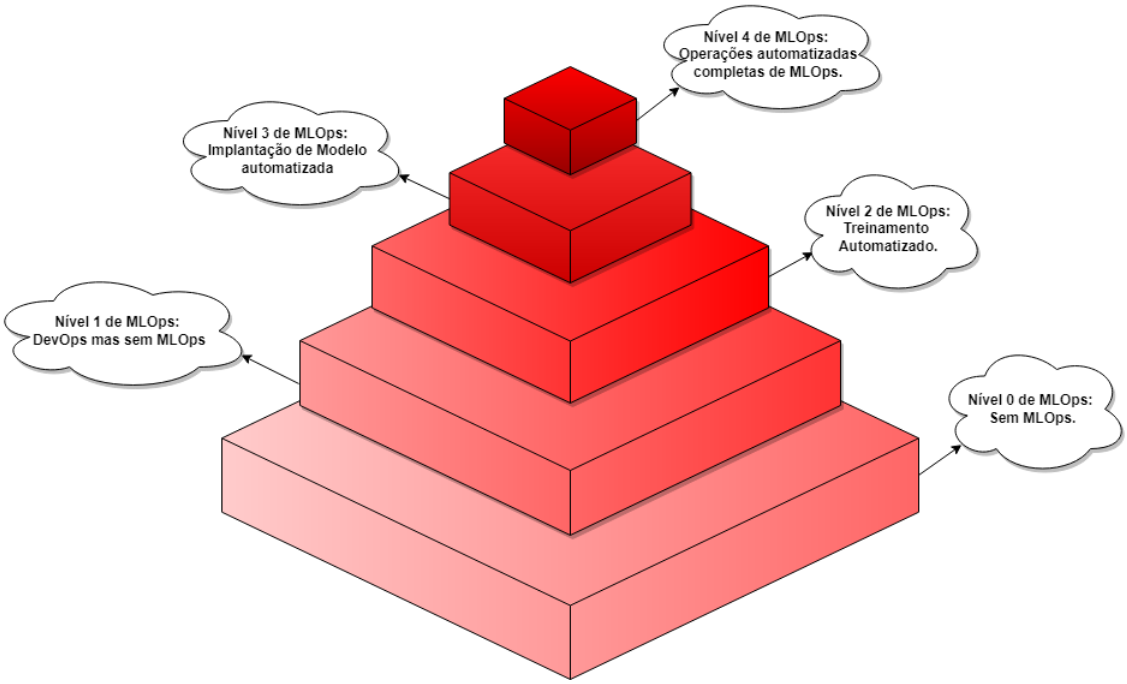

Figura 10 – Níveis de Maturidade da Microsoft. Fonte: De autoria própria

No modelo da Microsoft (MICROSOFT, 2022) existem cinco níveis de maturidade. Sua estrutura está apresentada na Figura 10.

Nível 0 - Sem MLOps. Tudo manual, extremamente parecido com o nível 0 da Google.

Nível 1- DevOps mas sem MLOps. Aqui já se define uma categoria para empresas ou times que seguem a cultura do DevOps, mas não aplicam os diferenciais da cultura do MLOps.

Nível 2 - Treinamento automatizado. Treinar modelos se torna uma tarefa simples. É possível criar vários modelos facilmente sem perder a rastreabilidade dos mesmos, mas ainda não é possível o treinamento contínuo, pois implantar o modelo ainda é uma tarefa manual.

Nível 3 - Implementação de Modelos automatizada. Enviar modelos para o ambiente de produção se torna uma tarefa automatizada e sem grandes dificuldades.

Nível 4 - Operações automatizadas completas de MLOps. Nesse caso, o MLOps estaria completo e automatizado. Existe o treinamento contínuo e é fácil experimentar e colocar novos modelos em produção.

Obviamente, todo time ou empresa gostaria de operar nos maiores níveis apresentados aqui, mas vale ressaltar que a transição entre os níveis não é obrigatoriamente discreta e que depende de cada projeto identificar suas necessidades.

4 ESTUDO DE CASO DA EMPRESA TAKE BLIP

4.1 Desenvolvimento

4.1.1 A empresa

Para realização do estudo de caso foi escolhido a empresa Take blip[1], uma empresa que trabalha com serviços e consultoria de TI. A empresa tem como visão criar contatos inteligentes que seriam uma versão aprimorada dos chatbots. É enquadrada na categoria de grande porte, com mais de 1400 funcionários[2], com sede na capital mineira, possuindo mais de 20 anos de história. A escolha da empresa se deu principalmente por três motivos:

1. Ser uma empresa que se autodenomina como *AI First*[3];
2. Possuir times com projetos experimentando o MLOps;
3. Facilidade de comunicação e coleta de informações.

4.1.2 O time

Dentro da Empresa, o time que cedeu o tempo e as informações está dentro da Diretoria de Pesquisa e Inovação e atende pela alcunha de Maestro. Esse time tem como propósito oficial "Construir e disponibilizar tecnologias utilizando inteligência artificial a partir de diferentes tipos de multimídia para melhorar a experiência do usuário, revolucionando a maneira como as marcas se comunicam com seus clientes." E para isso, utiliza extensivamente de modelos de aprendizado de máquina para a tarefa de processamento de linguagem natural ou *Natural Language Processing* (NLP) (VIEIRA; LOPES, 2010). O time é pequeno, possuindo apenas quatro integrantes até a finalização do presente trabalho. É composto por dois engenheiros de machine learning [4], uma cientista de dados e um estagiário, sendo os dois primeiros formados em áreas da tecnologia (Ciência da computação e Engenharia de Controle e Automação) e responsáveis pela engenharia das aplicações, construção das APIs e operação. A cientista de dados possui formação em Letras e tem papel de especialista em linguagem e é responsável por construir as bases de dados, modelos e heurísticas para a aplicação. Apesar das divisões anteriores o time age em conjunto e todos acabam contribuindo para a conclusão de todas as tarefas. Na Tabela 1 estão disponíveis algumas informações acerca dos 4 integrantes que compõem o time Maestro.

[1] Disponível pelo site: https://www.take.net/
[2] Informação retirada do Linekdin: https://www.linkedin.com/company/takeblip/
[3] Uma empresa *AI-First* é aquela que que possui um modelo de negócio desenhado para trabalhar exclusivamente com a tecnologia de Inteligencia Artificial afim de aumentar sua receita e ganhar espaço no mercado.
[4] O cargo de engenheiro de machine learning é comumente encontrado escrito dessa forma, sendo rara sua versão completamente traduzida para engenheiro de aprendizado de máquina.

4.2 Metodologia

Esta pesquisa utiliza do método de estudo de caso, para o qual Ventura (2007, p. 358) deu a seguinte definição:

> Como qualquer pesquisa, o estudo de caso é geralmente organizado em torno de um pequeno número de questões que se referem ao como e ao porquê da investigação. É provável que questões como essas estimulem também o uso de experimentos e pesquisas históricas.

Para realização deste estudo foram seguidos os seguintes passos: (i) Leitura de um documento realizado pela empresa sobre o atual estado da operacionalização de softwares alimentos por IA; (ii) Conversas com o time - a Tabela 1 ilustra os cargos dos integrantes do time; (iii) Coleta de informações com um integrante, identificado pela letra 'D' na Tabela 1; (iv) Síntese das informações coletadas e construção de um fluxo de informações que adéque aos conteúdos apresentados neste artigo.

Tabela 1 – Tabela com informações do time Maestro dentro da empresa Take Blip. Em azul esta destacada a linha com informações do integrante que proveu maior suporte para realização da pesquisa. T/L: Team Leader, TL: Tech Lead, MLE: Engenheiro de Machine Learning, DS: Cientista de Dados.

Integrante	Cargo	Descrição	Formação
A	T/L, MLE	Líder de pessoas do time, é também um Engenheiro de Machine Learning Sênior	Ciência da Computação
B	DS	Cientista de Dados do time, especialista em Português.	Letras
C	TL, MLE	Líder Técnico, é também um Engenheiro de Machine Learning Pleno	Engeharia de Controle e Automação
D	MLE	Estagiário de Engenheiro de Machine Learning	Estudante de Engenharia de Controle e Automação

4.3 Estado inicial do MLOps na empresa

Em um primeiro momento foi disponibilizado para a pesquisa um documento desenvolvido pelo time, que tinha como premissa analisar o atual estado do MLOps nos diversos projetos não só do time, como também de outros (com a única limitação de que o projeto abordado precisava utilizar de modelos de ML). O documento pontua o nível de maturidade do MLOps da empresa como sendo de 0 segundo a escala de maturidade da Google (GOOGLE, 2020), pois todos os passos, desde a criação do modelo, avaliação, testes até a implementação do modelo, eram feitas de forma manual. Os dados não eram versionados e muita das vezes estavam em repositórios privados ou na máquina pessoal do desenvolvedor, tornando impossível identificar qual conjunto de dados gerou qual modelo, o que dificultava reproduzir os resultados.

Outra característica importante é a frequência com a qual novos modelos eram treinados. No caso do time Maestro o re-treino acontecia poucas vezes ao ano por conta do tempo que isso acarretaria na esteira de desenvolvimento. Assim, implementar práticas do MLOps parecia cada vez mais uma necessidade.

A empresa já possuía uma cultura de DevOps bem desenvolvida, portanto a melhoria contínua em softwares se dava de forma extremamente veloz e robusta. Entretanto, o MLOps, por ser praticamente inexistente, fazia com que os modelos de aprendizado de máquina se tornassem gargalos do desenvolvimento. Na Figura 11 é possível perceber a falta de automatização do fluxo. Os dados não são automaticamente coletados por um pipeline de dados como o da Figura 8 indicado por 'B2' e sua extração e análises são feitas manualmente de repositórios da empresa (cilindro azul claro no topo da Figura). Posteriormente os dados passam pela etapas de experimentação manual. Apesar de ser comum existir essas etapas em pipelines de MLOps como mostrado no da Figura 8, no caso do time não existiam componentes prontos para cada etapa, ficando a cargo do desenvolvedor realiza-las manualmente toda vez que necessário. O artefato da experimentação é um modelo treinado que é armazenado no Cloud Storage e é então integrado no código da aplicação que irá para produção finalizando o pipeline. Não existe qualquer coleta de metadados e versionamento.

Outro problema pertinente ao fato de mudar do modo tradicional de desenvolvimento de modelos para um centrado na cultura do MLOps em uma empresa que já opera há tanto tempo, é a dificuldade em implementar todos os fluxos apresentados nos trabalhos de Symeonidis et al. (2022), Hewage e Meedeniya (2022) e Kreuzberger, Kühl e Hirschl (2022) sem que haja grandes atritos ou mudanças na infraestrutura já existente.

Como um ponto positivo, é possível ressaltar a colaboração e dinamismo do time. De acordo com a Tabela 1 é possível ver que os cientistas de dados responsáveis pelo desenvolvimento e os engenheiros de machine learning responsáveis pela operação já estão trabalhando em conjunto em um mesmo time, sendo esta a principal finalidade da cultura do MLOps, que é aproximar os times de desenvolvimento e operação.

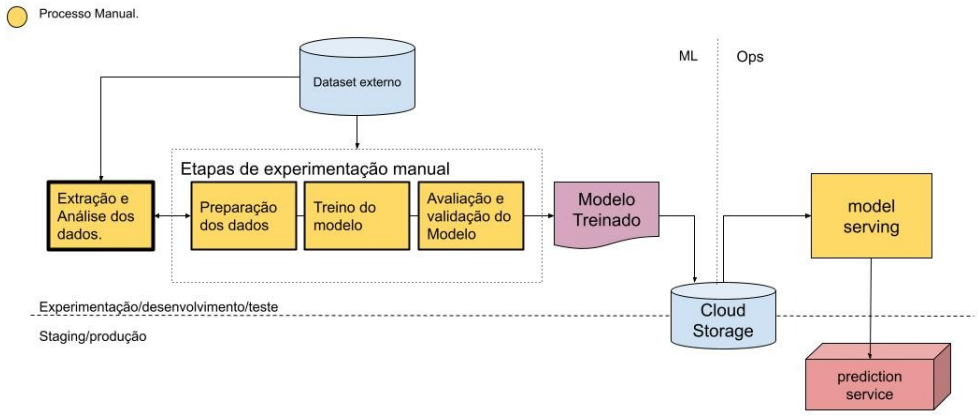

Figura 11 – Diagrama de alto nível da arquitetura inicial do MLOps do projeto. Fonte: Documentos internos do time Maestro.

4.4 Desenvolvimento

Conforme relatado pelo T/L (Tabela 1), o objetivo do time Maestro ao incorporar princípios do MLOps era acelerar a evolução dos modelos. Para isso, deveria ser criado um ferramental, no menor tempo possível, que possibilitaria aos cientistas de dados evoluir os modelos rapidamente e de uma forma mais simples.

Portanto, o primeiro passo que pensaram era automatizar a validação dos modelos. Validar um modelo pode ser visto de diversas formas. Nesse caso específico o time precisava de um método rápido que pudessem usar para aferir a qualidade em qualquer base de dados. Simplificadamente, a ideia era saber se o modelo estava performando de forma aceitável para bases de dados de novos clientes, gerando assim uma necessidade de também versionar estes dados. Inclusive, Shankar et al. (2022) expõem versionamento, velocidade e validação como as três características que ditam o sucesso para a implementação de modelos de ML em produção.

O versionamento foi introduzido com o auxilio da ferramenta DVC[5] (Data Version Control) e garantiu a possibilidade de saber exatamente qual resultado cada validação gerou. Além das bases de dados, os modelos também passaram a ser versionados, o que tornou mais fácil saber a melhor versão de um modelo para determinado cliente, por exemplo.

Até então o time tinha realizado o testes em um modelo específico, porém foi rapidamente constatado o aumento na velocidade e qualidade do trabalho com a introdução do componente automatizado de validação. O time então seguiu para criar uma ferramenta que iria automatizar a validação de uma aplicação por completo. A aplicação é composta por 4 modelos e o objetivo era coletar métricas de qualidade da mesma. Da mesma forma que antes, a ideia era deixar mais rápido e fácil a evolução da aplicação por completo, tornando possível fazer análises acerca da qualidade dos resultados para diferentes conjuntos de dados ou modelos. A Tabela 2 exemplifica a saída do componente de avaliação para a aplicação considerando diferentes bases de dados, e também é possível ver que o componente é capaz de testar para diferentes bases de dados ao mesmo tempo e informar diferentes métricas, que posteriormente seriam usadas pelo time para tomadas de ações como treinar novos modelos ou colocar o existente em produção.

Todavia, o treinamento dos modelos permanecia como uma tarefa manual. De acordo com o líder, o time sofre com o problema de que modelos são extremamente grandes e treiná-los demanda bastante tempo, chegando a durar vários dias. Além disso, ele conta que construir fluxos automatizados de treinamento contínuo não é tão simples e que apesar de empresas como Google, Amazon, Microsoft entre outras gigantes da tecnologia serem capazes de fazê-lo, é extremamente caro e trabalhoso para empresas como a Take blip.

[5] Disponível em: dvc.org

Tabela 2 – Exemplo da saída do componente de avaliação. Todos os valores são fictícios e não têm correlação com os da empresa.

Dataset	Métrica 1	Métrica 2	Métrica 3	Métrica 4
A	100	0,25	0,7	X
B	200	0,25	0,8	Y
C	300	0,1	0,96	X
Total	600	0,6	2,46	-
Média	200	0,2	0,82	-
Desvio Padrão	100	0,0866	0.1311	-

4.4.1 Percalços e projetos futuros

Além do problema citado anteriormente do tamanho dos modelos e a demora em treiná-los, outros problemas também apareceram durante a implementação. De acordo com o time algumas das ideias testadas, inclusive para validação dos dados, precisariam ser revistas por acarretarem problemas em desenvolvimentos posteriores.

Atualmente, a aplicação feita pelo time é de proposito genérico e usada em qualquer circunstância. Isto garante que apenas uma versão dos modelos que a compõem estejam em produção por vez. Entretanto é sabido que no futuro varias réplicas da aplicação com modelos diferentes serão necessárias e nesse instante, ter um operacionalização de ML robusta será extremamente necessária e benéfica.

De acordo com o Tech Lead (TL) (Tabela 1), pensar novas soluções para a engenharia do software e dos modelos é uma das tarefas que será desenvolvida no decorrer dos próximos meses e espera que ao final desse ciclo, o time seja capaz de criar modelos e coloca-los em produção de forma muito mais ágil e eficiente do que a que fazem agora.

5 DISCUSSÕES

As aplicação alimentadas por modelos de aprendizado de máquina são comumente mais difíceis de desenvolver, manter e dar manutenção se comparadas às tradicionais. Além da inserção de novos componentes para integrar os modelos de aprendizado de máquina ao serviço, tais componentes possuem as mesmas necessidades de serem testados e de garantia de qualidade que os tradicionais. Entretanto, tais necessidades não podem ser supridas por metodologias existentes e por isso faz-se necessário que a academia, industria e a comunidade criem novas ou adaptem as já difundidas, de modo a assegurar compatibilidade com essas aplicações emergentes na industria.

Após execução das pesquisas e revisão da literatura é possível traçar um paralelo entre a metodologia apresentadas por Studer et al. (2021), os conceitos desenvolvidos de DevOps e os princípios apresentados no capítulo 3 deste trabalho. O MLOps é visto por muitos como uma evolução ou ramificação do DevOps, o que explica o porquê de existirem conceitos parecidos ou iguais entre eles. O próprio nome é um exemplo, ambos são formados a partir da junção da palavra operação com o que precisava ser operacionalizado (desenvolvimento no caso do DevOps e Aprendizado de Máquina no do MLOps). Além disso, operação de aprendizado de máquina adiciona novos conceitos como CT (Treinamento contínuo) aos princípios de CI/CD e à necessidade de se testar e garantir a qualidade dos dados e modelos além da qualidade do código.

Agilidade e qualidade são sempre o foco das metodologias atuais, mas tais metodologias são extremamente teóricas e os desenvolvedores acabam caindo no problema de que "é mais fácil falar do que fazer". A cultura do DevOps sozinha não descreve com exatidão o que é preciso exercer de fato ao trabalhar com dados e modelos. Foi então que dessa necessidade dos times, surgiram modelos de processo como o CRISP-ML(Q) que discutem tecnicamente os passos para testar e garantir a qualidade quando se está criando serviços de aprendizado de máquina.

O MLOps surge com o aparente pretexto de conciliar todos os esforços relacionados à produção e operação de aprendizado de máquina sob uma mesma alcunha, de forma a facilitar a troca de informações e padronizar essa cultura emergente. Sendo assim, todos os times ou desenvolvedores que precisem operacionalizar modelos de ML possuem um mesmo ponto de partida na literatura.

Como no caso do time Maestro, onde existia uma enorme necessidade em adequar seu fluxo de melhoria dos modelos e inevitavelmente usaram de princípios do MLOps para isso. Apesar de não construir um pipeline completo, o estudo de caso exemplifica perfeitamente que pensar em MLOps e introduzir para dentro do time alguns de seus princípios é capaz de garantir resultados significativos. O time tornou-se mais dinâmico e incisivo em suas entregas e conseguiu adquirir velocidade e maturidade no aperfeiçoamento dos modelos.

Entretanto, após comparar o que foi feito pelo time em sua tentativa de migrar para um desenvolvimento suportado pela cultura de MLOps, percebe-se que eles estão longe de cumprir todos os requisitos do pipeline apresentado pela Figura 8. Isto demonstra, na opinião do autor, o principal ponto negativo da cultura: criar e manter um pipeline tão complexo é extremamente difícil e demanda um conhecimento técnico elevado das equipes de desenvolvimento e operação, especialmente em empresas que pretendem fazê-lo depois de já consolidadas no mercado. O time Maestro não podia simplesmente abandonar a engenharia existente e construir uma nova pois ia lhes custar muito tempo e dinheiro para conceber algo que era e ainda é incerto.

Apesar do problema acima, o Maestro mostrou resultados excelentes ao introduzir algumas melhorias no pipeline de desenvolvimento focadas em problemas momentâneos. O time elencou os principais pontos de ajustes e partir daí criaram componentes e automações justamente onde seriam mais beneficiados. Apesar disso, do ponto de vista do autor do trabalho, existe uma necessidade de desacoplar os modelos de aprendizado de máquina da aplicação que os consomem. Algo como uma aplicação separada que funciona apenas para servir um modelo. Ao fazer isso será possível criar novas versões de modelos sem mexer no código da aplicação o que torna o pipeline de desenvolvimento mais simplificado e fácil de automatizar.

Entretanto com o tempo, as empresas, principalmente as gigantes da tecnologia, vão criando os padrões e a gigantesca quantidade de ferramental que está atualmente difundida no mercado vai minguando mais e mais à medida que os times descobrem o que é necessário e o que não é. Isto se estende também aos pipelines e é possível que estes se tornem mais simples à medida que o tempo passe ou pelo menos mais fáceis de entender.

6 CONSIDERAÇÕES FINAIS

A presente monografia tem como objetivo principal apresentar uma revisão do tema MLOps, de forma a introduzir o assunto para leigos e pontuar os principais pontos desta cultura. Tal objetivo é até certo ponto alcançado, pois durante a leitura do trabalho é possível tomar conhecimento do assunto e entender os principais conceitos da metodologia bem como suas diferenças e similaridades com metodologias já existentes. A pesquisa sintetiza os conceitos introdutórios e serve como material teórico para todos que desejam entender melhor o tema.

Como conclusão do autor, a grande quantidade de discussões acerca do assunto e o crescente número de ferramentas sendo criadas para as mais diversas etapas do pipeline (SYMEONIDIS et al., 2022), sem que haja ainda um padrão aceito tanto na industria quanto na academia, acaba por se tornar um agravante ao invés de um atenuante. Aqueles que buscam pelas ferramentas ou guias se deparam com soluções que talvez funcionem para casos específicos, mas não para todos de modo geral.

É também possível concluir, através do estudo de caso, que a complexidade em adotar-se as técnicas apresentadas é alta, além de demandar um elevado nível técnico e de cooperação entre os integrantes de um time e os responsáveis pela operacionalização. Mesmo entre as maiores empresas não existe consenso quanto aos passos a serem tomados e o que deve ser executado de fato, como é possível constatar ao comparar as diferentes estruturas dos níveis de maturidade apresentados em (MICROSOFT, 2022) e (GOOGLE, 2020) . Porém, o time estudado demonstra que, apesar de já terem adotado plenamente um modo de pensar centrado nos conceitos do MLOPs, ainda irá levar um tempo até que essa mudança seja completamente difundida na prática. É possível perceber que pequenas interações de evolução, priorizando componentes e etapas que garantam o maior retorno em curto prazo, são extremamente eficazes quando se está criando um pipeline de aprendizado de máquina. Não sendo necessário implementar em uma única interação um pipeline tão complexo quanto o proposto por Kreuzberger, Kühl e Hirschl (2022).

Finalmente, conclui-se que o MLOps é extremamente novo na mente de todos e está em seus estágios inicias, portanto ainda irá amadurecer com o tempo. As ferramentas e aplicações que surgem aos montes no mercado irão aos poucos desaparecer, restando apenas as consideradas padrão pela comunidade como foi no caso do DevOps. Além disso, o presente trabalho não representa um fim para a discussão sobre o assunto, sendo de fato o exato oposto - uma introdução a uma discussão interminável de como colocar em produção os emergentes softwares alimentados por modelos de aprendizado de máquina.

6.1 Trabalhos Futuros

Em cima do exposto, o autor propõem os seguintes trabalhos futuros:

- Proposição de um pipeline de MLOps para um caso de exemplo.
- Desenvolvimento prático de um pipeline.
- Comparação entre um desenvolvimento tradicional para um centrado nos conceitos do MLOps quanto aos benefícios e malefícios encontrados.

REFERÊNCIAS

AKEN, B. van et al. *Clinical Outcome Prediction from Admission Notes using Self-Supervised Knowledge Integration*. arXiv, 2021. Disponível em: <https://arxiv.org/abs/2102.04110>. Citado na página 15.

ALPERSTEIN, Z.; CHERKASOV, A.; ROLFE, J. T. *All SMILES Variational Autoencoder*. arXiv, 2019. Disponível em: <https://arxiv.org/abs/1905.13343>. Citado na página 15.

ARORA, A. et al. Hybrid android malware detection by combining supervised and unsupervised learning. In: *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: Association for Computing Machinery, 2018. (MobiCom '18), p. 798–800. ISBN 9781450359030. Disponível em: <https://doi.org/10.1145/3241539.3267768>. Citado na página 18.

BECK, K. et al. Manifesto for agile software development. 2001. Snowbird, UT, 2001. Citado 2 vezes nas páginas 15 e 19.

BENčEVIć, M. et al. Training on polar image transformations improves biomedical image segmentation. *IEEE Access*, 2021. v. 9, p. 133365–133375, 2021. Citado na página 15.

DAGLI, R.; SHAIKH, A. M. *CPPE-5: Medical Personal Protective Equipment Dataset*. arXiv, 2021. Disponível em: <https://arxiv.org/abs/2112.09569>. Citado na página 15.

FORET, P. et al. *Sharpness-Aware Minimization for Efficiently Improving Generalization*. arXiv, 2020. Disponível em: <https://arxiv.org/abs/2010.01412>. Citado na página 15.

GÉRON, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. [S.l.]: "O'Reilly Media, Inc.", 2019. Citado na página 17.

GHIASI, G. et al. *Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation*. arXiv, 2020. Disponível em: <https://arxiv.org/abs/2012.07177>. Citado na página 15.

GOOGLE. *MLOps: pipelines de entrega contínua e automação no aprendizado de máquina*. 2020. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning. Citado 3 vezes nas páginas 29, 33 e 39.

HASSAN, M. R.; NATH, B.; KIRLEY, M. A fusion model of hmm, ann and ga for stock market forecasting. *Expert Systems with Applications*, 2007. v. 33, n. 1, p. 171–180, 2007. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417406001291>. Citado na página 23.

HENDRYCKS, D. et al. *Measuring Mathematical Problem Solving With the MATH Dataset*. arXiv, 2021. Disponível em: <https://arxiv.org/abs/2103.03874>. Citado na página 15.

HEWAGE, N.; MEEDENIYA, D. Machine learning operations: A survey on mlops tool support. 2022. arXiv, 2022. Disponível em: <https://arxiv.org/abs/2202.10169>. Citado 4 vezes nas páginas 22, 25, 26 e 34.

INSTUTUTE, S. S. T. S. *All QUick Facts*. 2021. https://webbtelescope.org/quick-facts/all-quick-facts. Citado na página 14.

IPPOLITO, M.; FERGUSON, J.; JENSON, F. Improving facies prediction by combining supervised and unsupervised learning methods. *Journal of Petroleum Science and Engineering*, 2021. v. 200, p. 108300, 2021. ISSN 0920-4105. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0920410520313541>. Citado na página 18.

JIANG, H. et al. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020. Disponível em: <https://doi.org/10.18653%2Fv1%2F2020.acl-main.197>. Citado na página 15.

KABIR, H. M. D. et al. *SpinalNet: Deep Neural Network with Gradual Input*. arXiv, 2020. Disponível em: <https://arxiv.org/abs/2007.03347>. Citado na página 15.

KREUZBERGER, D.; KÜHL, N.; HIRSCHL, S. Machine learning operations (mlops): Overview, definition, and architecture. *arXiv preprint arXiv:2205.02302*, 2022. 2022. Citado 7 vezes nas páginas 15, 19, 22, 25, 27, 34 e 39.

MICROSOFT. *Machine Learning operations maturity model*. 2022. https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model. Citado 2 vezes nas páginas 30 e 39.

MLOPS-SIG. *MLOps Roadmap 2022 - DRAFT COPY*. 2022. https://github.com/cdfoundation/sig-mlops/blob/main/roadmap/2022/MLOpsRoadmap2022.md. Citado 2 vezes nas páginas 19 e 22.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *Sistemas inteligentes-Fundamentos e aplicações*, 2003. Manole, v. 1, n. 1, p. 32, 2003. Citado na página 14.

PALEYES, A.; URMA, R.-G.; LAWRENCE, N. D. Challenges in deploying machine learning: A survey of case studies. *ACM Comput. Surv.*, 2022. Association for Computing Machinery, New York, NY, USA, apr 2022. ISSN 0360-0300. Just Accepted. Disponível em: <https://doi.org/10.1145/3533378>. Citado na página 24.

PERERA, P.; SILVA, R.; PERERA, I. Improve software quality through practicing devops. In: . [S.l.: s.n.], 2017. p. 1–6. Citado 2 vezes nas páginas 18 e 19.

RAFFEL, C. et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. arXiv, 2019. Disponível em: <https://arxiv.org/abs/1910.10683>. Citado na página 15.

ROYCE, W. W. Managing the development of large software systems: concepts and techniques. In: *Proceedings of the 9th international conference on Software Engineering*. [S.l.: s.n.], 1987. p. 328–338. Citado na página 15.

SCULLEY, D. et al. Hidden technical debt in machine learning systems. *Advances in neural information processing systems*, 2015. v. 28, 2015. Citado na página 22.

SHANKAR, S. et al. Operationalizing machine learning: An interview study. *arXiv preprint arXiv:2209.09125*, 2022. 2022. Citado 2 vezes nas páginas 22 e 35.

SHOEYBI, M. et al. *Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism*. arXiv, 2019. Disponível em: <https://arxiv.org/abs/1909.08053>. Citado na página 15.

STUDER, S. et al. Towards crisp-ml (q): a machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction*, 2021. Multidisciplinary Digital Publishing Institute, v. 3, n. 2, p. 392–413, 2021. Citado 4 vezes nas páginas 20, 21, 25 e 37.

SYMEONIDIS, G. et al. Mlops - definitions, tools and challenges. In: *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*. [S.l.: s.n.], 2022. p. 0453–0460. Citado 5 vezes nas páginas 22, 25, 26, 34 e 39.

TEIXEIRA, J. *O que é inteligência artificial*. [S.l.]: E-Galáxia, 2019. Citado na página 14.

TESTI, M. et al. Mlops: A taxonomy and a methodology. *IEEE Access*, 2022. v. 10, p. 63606–63618, 2022. Citado na página 22.

TOUVRON, H. et al. *Going deeper with Image Transformers*. arXiv, 2021. Disponível em: <https://arxiv.org/abs/2103.17239>. Citado na página 15.

VENTURA, M. M. O estudo de caso como modalidade de pesquisa. *Revista SoCERJ*, 2007. v. 20, n. 5, p. 383–386, 2007. Citado 2 vezes nas páginas 15 e 33.

VIEIRA, R.; LOPES, L. Processamento de linguagem natural e o tratamento computacional de linguagens científicas. *Em corpora*, 2010. p. 183, 2010. Citado na página 32.

WIRTH, R.; HIPP, J. Crisp-dm: Towards a standard process model for data mining. In: MANCHESTER. *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. [S.l.], 2000. v. 1, p. 29–39. Citado na página 20.

YU, J. et al. *CoCa: Contrastive Captioners are Image-Text Foundation Models*. arXiv, 2022. Disponível em: <https://arxiv.org/abs/2205.01917>. Citado na página 15.

ZHANG, G.; XU, L.; XUE, Y. Model and forecast stock market behavior integrating investor sentiment analysis and transaction data. *Cluster Computing*, 2017. Springer, v. 20, n. 1, p. 789–803, 2017. Citado na página 23.

ZHANG, H. et al. *DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection*. arXiv, 2022. Disponível em: <https://arxiv.org/abs/2203.03605>. Citado na página 15.

ZHENG, A. et al. *Progressive End-to-End Object Detection in Crowded Scenes*. arXiv, 2022. Disponível em: <https://arxiv.org/abs/2203.07669>. Citado na página 15.

ZHOU, C. et al. One-pass multi-task networks with cross-task guided attention for brain tumor segmentation. *IEEE Transactions on Image Processing*, 2020. Institute of Electrical and Electronics Engineers (IEEE), v. 29, p. 4516–4529, 2020. Disponível em: <https://doi.org/10.1109%2Ftip.2020.2973510>. Citado na página 15.