UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Instituto de Ciência e Tecnologia
Câmpus de Sorocaba

SEIRYO KASHIWABA

OTIMIZAÇÃO DE VENDAS NO SETOR VAREJISTA COM INTELIGÊNCIA ARTIFICIAL

Sorocaba
2025

SEIRYO KASHIWABA

OTIMIZAÇÃO DE VENDAS NO SETOR VAREJISTA COM INTELIGÊNCIA ARTIFICIAL

Trabalho de Conclusão de Curso apresentado à Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba, como parte dos requisitos para obtenção do grau de Bacharel(a) em Engenharia de Controle e Automação.

Orientador: Prof. Dr. Everson Martins

Coorientador: Prof. Dr. Ivando Severino Diniz

Sorocaba

2025

K19o

Kashiwaba, Seiryo
Otimização de vendas no setor varejista com inteligência artificial / Seiryo Kashiwaba. -- Sorocaba, 2025
89 p. : tabs., fotos

Trabalho de conclusão de curso (Bacharelado - Engenharia de Controle e Automação) - Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba
Orientador: Everson Martins
Coorientador: Ivando Severino Diniz

1. Aprendizado do computador. 2. Redes neurais (Computação). 3. Inteligência artificial. I. Título.

SEIRYO KASHIWABA

OTIMIZAÇÃO DE VENDAS NO SETOR VAREJISTA COM INTELIGÊNCIA ARTIFICIAL

Trabalho de Conclusão de Curso apresentado à Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba, como parte dos requisitos para obtenção do grau de Bacharel em Engenharia de Controle e Automação.

Data da defesa: 12/03/2025

BANCA EXAMINADORA:

gov.br
Documento assinado digitalmente
EVERSON MARTINS
Data: 13/03/2025 18:24:43-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Everson Martins
UNESP - Instituto de Ciência e Tecnologia - Campus de Sorocaba

gov.br
Documento assinado digitalmente
IVANDO SEVERINO DINIZ
Data: 13/03/2025 14:52:30-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Ivando Severino Diniz
UNESP - Instituto de Ciência e Tecnologia - Campus de Sorocaba

gov.br
Documento assinado digitalmente
FELIPE RIBEIRO DE MELLO FERREIRA
Data: 12/03/2025 15:22:05-0300
Verifique em https://validar.iti.gov.br

Eng. Felipe Ribeiro de Mello Ferreira
Membro Externo

AGRADECIMENTOS

Ao meu falecido pai, Tokio Kashiwaba, que acreditou em meus estudos, desde a minha ida a uma escola particular de elite, apostando em um futuro melhor, possibilitando eu cursar Engenharia em uma das melhores universidades de São Paulo. À minha mãe, Hisae Yoshida Kashiwaba, que sempre me cuidou em tempos difíceis.

À minha namorada, Michelle Yuka Takahashi, que praticamente viu toda a minha trajetória acadêmica e profissional. Sempre me apoiou e me motivou com seu esforço, pois nunca vi alguém tão batalhadora como ela.

Ao meu orientador, Prof. Dr. Everson Martins, que além de professor, foi um grande companheiro de Congregação quando fui presidente do Centro Acadêmico, sempre me auxiliando em todo o contexto da UNESP. Me ajudou muito nesse grande desafio que tive na minha vida.

Aos meus irmãos Yoshihide, Miwa, Yoshitaka, Yoshikazu e Yoshitada que sempre serão a maior base de apoio da minha vida, pois sem eles, eu não teria nenhuma personalidade de quem eu sou hoje. Todas as minhas virtudes vieram de vocês.

Aos meus amigos da República Txeca e em especial ao Cagão, Guita, Yudi, Judas, Mandela e Tiago, que sem vocês, a minha vida acadêmica iria ser muito mais difícil. Obrigado por termos muitas lembranças juntos, foram um dos melhores momentos da minha vida e jamais será esquecido.

Ao meu amigo Felipe Mello, que me inseriu no mercado de trabalho e hoje é o meu maior companheiro de equipe. Sem ele, não teria sido possível realizar esse trabalho pois jamais iria ter experiência com inteligência artificial. Aprendi muito com o senhor, só tenho agradecimentos.

Aos meus amigos de São Paulo, “grupo dos trabalhadores da Vivo”, que fazem o meu dia a dia sempre melhor. Agradeço desde sempre e esperamos ter um laço até o fim das nossas vidas.

RESUMO

O setor varejista está sofrendo muitas mudanças após o “boom” da inteligência artificial, nesse setor, há a área de *Customer Relationship Management* (CRM), a qual realiza gestão de relacionamento que visa melhorar a satisfação, otimizar a rentabilidade, aumentar a receita e visar a assertividade das campanhas de captação de novos clientes. Este projeto visa aplicar inteligência artificial nesse ramo. Na revisão de literatura desta tese, primeiramente, foram abordados os seguintes conceitos: *Customer Relationship Management*, para o entendimento de onde está sendo aplicado a inteligência artificial; *Machine Learning*, explicando o que é, quais são suas aplicações, como realizar as coletas de dados, como preparar os dados com as técnicas de *feature engineering* e como realizar o treinamento do modelo; Utilização da ferramenta poderosa de análise de dados da Google, BigQuery, explicando o que é e como utilizá-la, para treinar o modelo para realizar predições dos *scores* dos consumidores; Percentil, pois com as predições realizadas, é necessário realizar a classificação dos consumidores, onde acontecerá uma ordenação de *scores* e classificações baseado na posição relativa; Teste A/B, técnica de realizar experimentos separando em grupos de controle e variante; Visualização de dados, explicando os tipos de gráficos e quando usá-los; Dito CRM, um software utilizado para realizar disparos de campanhas com Inteligência Artificial. Com os disparos feitos para o grupo de controle e grupo de variante, foi verificado que o grupo com a aplicação de inteligência artificial obteve resultados de quase quatro vezes superior em relação ao grupo de controle.

Palavras-chave: Aprendizado de Máquinas; Redes Neurais; Teste A/B; Visualização de Dados; Gestão de Relacionamento com o Cliente.

ABSTRACT

The retail sector is undergoing significant changes after the “boom” of artificial intelligence. Within this sector, there is the area of Customer Relationship Management (CRM), which manages customer relationships with the aim of improving satisfaction, optimizing profitability, increasing revenue, and enhancing the effectiveness of customer acquisition campaigns. This project aims to apply artificial intelligence in this domain. The literature review of this thesis first addresses the following concepts: Customer Relationship Management to understand where artificial intelligence is being applied; Machine Learning, explaining what it is, its applications, how to collect data, prepare data with feature engineering techniques, and train models; BigQuery, Google's powerful data analytics tool, explaining what it is and how to use it to train models for consumer score predictions; Percentile, as consumer predictions require classification by ranking scores and assigning classifications based on relative positions; A/B Testing, a technique for conducting experiments by separating data into control and variant groups; Data Visualization, explaining the types of charts and when to use them; and Dito CRM, a software tool used to launch AI-driven campaigns..With the messages delivered to the control group and the variant group, it was found that the group with the application of artificial intelligence achieved results nearly four times higher compared to the control group.

Keywords: Machine Learning; Neural Networks; A/B Testing; Data Visualization; Customer Relationship Management.

LISTA DE ILUSTRAÇÕES

Figura 1 – Representação visual da relação entre as áreas de dados ……..... 20
Figura 2 – Ciclo de vida de um projeto de Machine Learning 21
Figura 3 – Vazamento de dados de um modelo 23
Figura 4 – Particionamento da base de dados 24
Figura 5 – Exemplo de um dataset tabular 25
Figura 6 – Exemplo de um dataset tabular de línguas que podem ser descontinuadas …. 26
Figura 7 – Exemplo de um dataset tabular de um e-commerce sem mescla ……. 26
Figura 8 – Exemplo de um dataset tabular de um e-commerce com mescla ……. 27
Figura 9 – Exemplo de um dataset tabular de uma corrida de animais ……. 27
Figura 10 – Exemplo de um dataset tabular de línguas que podem ser descontinuadas com o seu potencial de extinção antes da aplicação do one-hot encoding …… 28
Figura 11 – Exemplo de um dataset tabular de línguas que podem ser descontinuadas com o seu potencial de extinção com aplicação do one-hot enconding …… 28
Figura 12 – Sobreamostragem (esquerda) e subamostragem (direita) …….. 30
Figura 13 – Rede neural biológica ……. 30
Figura 14 – Rede neural artificial …….. 31
Figura 15 – Perceptron …….. 32
Figura 16 – Pré e pós ativação dos valores com o neurônio …….. 33
Figura 17 – Arquitetura básica de uma rede feed-forward com duas camadas intermediárias e uma única camada de saída ……. 34
Figura 18 – Simples computação gráfica com dois nós …… 35
Figura 19 – Simples computação gráfica com dois nós intermediários …… 36
Figura 20 – Computação gráfica com cinco nós intermediários ……... 37
Figura 21 – A influência da taxa de aprendizado para a convergência …… 40
Figura 22 – Área abaixo da curva ROC (AUC) mostrado em cinza ……. 45
Figura 23 – Console do BigQuery ……. 51
Figura 24 – Separação dos grupos em variante de controle e variante de teste …… 59
Figura 25 – Gráfico original das mães que ficam em casa …… 60
Figura 26 – Remodelação do gráfico original das mães que ficam em casa ……. 61
Figura 27 – Bordas de tabela ……. 61
Figura 28 – Diferenciação entre tabela e mapa de calor …… 62

Figura 29 – Custo por milha por milhas dirigidas ……. 62
Figura 30 – Custo por milha por milhas dirigidas com a linha da média …….. 63
Figura 31 – Tempo de espera no controle de passaportes ……. 63
Figura 32 – Opinião dos funcionários ao longo do tempo …… 64
Figura 33 – Gráficos de barra com e sem linha de base zero …… 65
Figura 34 – Gráficos de barras verticais …… 65
Figura 35 – Gráficos de barras empilhadas ……... 66
Figura 36 – Número de funcionários em gráficos de cascata …… 66
Figura 37 – Gráfico de barras com uma ou mais séries ……. 67
Figura 38 – Gráfico de barras horizontais empilhadas a 100% ……. 67
Figura 39 – Gráfico de área ……... 68
Figura 40 – Tela de segmentação dos consumidores no aplicativo da Dito …….. 68
Figura 41 – Segmentação com “Propensão de compra” ……... 69
Figura 42 – Opção de disparo de campanha …….. 69
Figura 43 – Diagrama do procedimento experimental do projeto …… 70
Figura 44 – Criação do modelo no GCP …… 73
Figura 45 – Dados do consumidor no aplicativo da Dito CRM …… 75
Figura 46 – Perdas do modelo e duração do treino no GCP …….. 76
Figura 47 – Avaliação do modelo no GCP …… 76
Figura 48 – Matriz de confusão do modelo no GCP ……. 77
Figura 49 – Resultado da predição do modelo no GCP …… 78
Figura 50 – Classificação das predições do modelo no GCP …… 79
Figura 51 – Transações e conversão das campanhas de SMS da variante de controle (à esquerda) e da variante de teste (à direita) ……. 80

LISTA DE QUADROS

Quadro 1 – Código da criação do modelo no BigQuery 52
Quadro 2 – Código da avaliação do modelo no BigQuery..... 56
Quadro 3 – Código da predição do modelo no BigQuery..... 56

LISTA DE TABELAS

Tabela 1 – Exemplo de matriz de confusão……..... 43

LISTA DE ABREVIATURAS E SIGLAS

IA	Inteligência Artificial
CRM	Customer Relationship Management
ML	Machine Learning
DNN	Deep Neural Networks
MSE	Mean Squared Error
MAE	Median Absolute Error
BCE	Binary Cross-Entropy

SUMÁRIO

1	**INTRODUÇÃO**	**13**
2	**JUSTIFICATIVA E OBJETIVOS**	**14**
3	**REVISÃO DE LITERATURA**	**15**
3.1	**Gestão de Relacionamento com o Cliente (CRM)**	**15**
3.1.1	Introdução ao CRM	15
3.1.2	Tipos de CRM	16
3.2	**Machine Learning (Aprendizado de Máquinas)**	**17**
3.2.1	Introdução ao Machine Learning	17
3.2.2	Preparação e coleção dos dados	21
3.2.3	Feature engineering	24
3.2.4	Redes Neurais	30
3.2.5	Treinamento do modelo	40
3.2.6	Passo a passo do treinamento	48
3.3	**BigQuery**	**49**
3.3.1	Introdução ao BigQuery	49
3.3.2	Organização dos recursos	50
3.3.2	BigQuery ML	52
3.4	**Percentis**	**57**
3.4.1	Conceito de percentil	57
3.4.2	O resumo dos cinco números	58
3.5	**Teste A/B**	**58**
3.6	**Visualização de dados**	**59**
3.6.1	Tipos de Gráficos	60
3.7	**Dito CRM**	**68**
4	**METODOLOGIA**	**70**
4.1	**Criação da feature store**	**70**
4.2	**Treinamento, avaliação e predição do modelo**	**72**
4.3	**Classificação dos consumidores com percentis**	**74**
4.4	**Disparo das notificações para os consumidores em grupos de controle**	**74**
5	**RESULTADOS**	**76**
5.1	**Resultado do treinamento do modelo**	**76**

5.2 Resultado das predições do modelo e classificações com percentil …..... 78
5.3 Disparo das notificações para os consumidores …….... 79
6 CONCLUSÃO 81
REFERÊNCIAS 82
APÊNDICE A – Código da criação do modelo no BigQuery ……..... 84
APÊNDICE B – Código da avaliação do modelo no BigQuery……..... 85
APÊNDICE C – Código da predição do modelo no BigQuery 86
APÊNDICE D – Código da classificação com percentis no BigQuery 88

1 INTRODUÇÃO

A ideia de inteligência artificial (IA) foi instaurada em 1950, por Alan Turing, indagando se é possível fazer as máquinas pensarem. Essa tecnologia sempre foi crescendo, ocorrendo um “boom” com a instauração do ChatGPT, ferramenta da OpenAI. A ferramenta se popularizou tanto que fez os humanos questionarem inúmeros fatores nos próximos anos. Segundo Anderson Rocha, pesquisador e coordenador da Universidade Estadual de Campinas (Unicamp), os principais benefícios dessas tecnologias vêm da utilização das IAs como um copiloto inteligente para realizar atividades técnicas. Por exemplo, no ramo de atendimento, a ferramenta é usada para realizar triagens para cada indivíduo, na programação, ela pode resolver códigos, na área da saúde, ela pode acelerar os resultados dos exames e entre outras aplicações do cotidiano (Veja, 2023).

Além dessas áreas citadas anteriormente, há o setor varejista, em que será amplamente discutido nesta tese. O setor tem aplicações para gestão de inventário, planejamento de sortimentos, marketing, merchandising e outras atividades comerciais. Com a IA, podem sugerir o reabastecimento de mercadorias de acordo com as necessidades, organizações de estoque, oferecer promoções de marketing personalizadas que sejam mais relevantes e oportunas para os consumidores e que visam o lucro (Oracle, 2024).

Sendo mais específico na área do varejo, existe a área de Customer Relationship Management (CRM), em que a estratégia de negócios centrada no cliente que visa melhorar a satisfação, otimizar a rentabilidade, aumentar a receita e visa a assertividade das campanhas de captação de novos clientes (Salesforce, 2016). A integração entre IA e CRM permite que o processo de vendas seja eficiente para identificar a melhor forma de atender cada cliente por meio da aplicação de deep learning com os dados dos consumidores (Salesforce, 2024).

Nesta tese, serão abordados os conceitos de: CRM; Machine Learning (ML); BigQuery (ferramenta analítica de Big Data); Feature Engineering (preparação dos dados para aplicações de ML); algoritmo das redes neurais; treinamento de modelos; classificação de consumidores por percentis; realização de experimentos com Testes A/B; geração de gráficos com técnicas de visualização de dados e, por fim, disparo de campanhas utilizando o software Dito CRM. Juntando todos esses conhecimentos, será possível realizar um projeto de otimização de vendas no setor varejista.

2 JUSTIFICATIVA E OBJETIVOS

Segundo o Instituto de Tecnologia de Massachusetts (MIT), as empresas podem se beneficiar de vantagens competitivas ao integrar soluções de inteligência artificial em seus negócios. Empresas orientadas a dados precisarão adquirir mais dados, pois elas são a matéria para que se tenha a inteligência de negócio capaz de impulsionar melhorias nas operações de negócios ou de conduzir ao surgimento de novas linhas de negócios (Exame, 2023).

Com o aumento da concorrência, as companhias estão procurando soluções tecnológicas que potencializam todos os seus serviços existentes. Este projeto tem como justificativa mostrar como a integração de IA com sistemas de CRM pode transformar a forma como as aplicações envolvem o comportamento do consumidor.

O objetivo principal desse projeto é utilizar Redes Neurais Profundas, ou Deep Neural Networks (DNN) com os comportamentos dos consumidores como variáveis de entrada para realizar classificações de probabilidades de compra. Com a base inteira classificada com as probabilidades, é possível agrupá-la, com o método estatístico percentil, em quatro grupos: “Alta Propensão”, “Média Propensão”, “Baixa Propensão” e “Sem Propensão”. Com os grupos formados, são feitos disparos de campanhas em clientes que são propensos e clientes sem classificação para o experimento do Teste A/B. Os resultados desse teste são colocados em um gráfico para permitir realizar conclusões sobre a efetividade do uso da inteligência artificial no mercado.

3 REVISÃO DE LITERATURA

Este capítulo foi separado em seções da seguinte forma:

- Seção 3.1: CRM para o entendimento do contexto da área de aplicação da IA
- Seção 3.2: Machine Learning para saber como uma IA é aprendida e o porquê funciona.
- Seção 3.3: Deep Neural Networks, para saber como um dos tipos de algoritmos funciona.
- Seção 3.4: BigQuery, ferramenta da Google de análise de dados para treinar, avaliar e realizar predições do modelo de redes neurais profundas.
- Seção 3.5: Percentil, para classificar as predições realizadas dos consumidores.
- Seção 3.6: Teste A/B, para realizar experimentos entre um grupo de controle e grupo variante.
- Seção 3.7: Visualização de Dados, para gerar gráficos para realizar conclusões do experimento do projeto.

O capítulo terá bastante embasamento teórico para que se possa ter o máximo entendimento de todos os conceitos para a realização do projeto. Nota-se que muitos termos serão escritos em inglês devido à natureza da área de tecnologia. Grande parte dos frameworks, softwares e ferramentas são desenvolvidos nesse idioma e como não existem traduções em português que mantenham a precisão técnica. O uso desses termos em inglês também pode familiarizar-se com o vocabulário do mercado global, além de que o mercado nacional na área de tecnologia costuma utilizar os termos em inglês também.

3.1 Gestão de Relacionamento com o Cliente (CRM)

3.1.1 Introdução ao CRM

A gestão de relacionamento com o cliente surgiu por volta do fim da década de 1990 como uma estratégia centrada ao redor da coleta e uso da informação entre as relações e transações. CRM é uma solução suplementar para as seguintes principais aplicações:

- Aplicações transacionais: aplicações usadas pelos encarregados com as atividades altamente relacionadas com o cliente, ou front office, para lidar com as transações entre a organização e o consumidor.
- Aplicações de gestão ou contabilidade: essas aplicações são usadas pelos encarregados com as atividades não tão relacionadas diretamente com o cliente, ou back office, para consolidar dados necessários para cumprir os deveres de

contabilidade e ajudar na área administrativa como finanças, logísticas e recursos humanos.

CRM é um sistema integrado com foco nos consumidores e potenciais consumidores, ou leads. Isso contém processos e procedimentos que constroem um modelo centrado de gestão em uma visão 360º do consumidor. O processo de gestão é baseado em níveis de estratégia, marketing, finanças e economia. A tecnologia pode ajudar na captura dos dados sobre o cliente via fontes externas e consolidar esses dados em um armazém de dados, ou data warehouse (Helgeon, 2017).

3.1.2 Tipos de CRM

Nesta subseção serão mostrados os quatro tipos de CRM, indicando suas características e seus usos.

- Estratégico

O CRM estratégico foca no desenvolvimento de uma cultura empresarial centrada no cliente. Essa cultura é dedicada a conquistar e manter clientes ao oferecer e entregar valor de forma superior à dos concorrentes. Ela se reflete no comportamento das lideranças, no design dos sistemas formais da empresa, e nas histórias e mitos criados internamente. Em uma cultura centrada no cliente, espera-se que os recursos sejam alocados para maximizar o valor ao cliente, que os sistemas de recompensas incentivem comportamentos que aumentem a satisfação e retenção de clientes, e que as informações sobre os clientes sejam coletadas, compartilhadas e aplicadas em toda a empresa.

A centralidade no cliente compete com outras lógicas de negócios. Philip Kotler identifica três outras principais orientações empresariais: orientada ao produto, à produção e à venda. As empresas orientadas ao produto acreditam que os clientes escolhem os produtos com a melhor qualidade, desempenho, design ou características. Essas empresas costumam ser inovadoras e empreendedoras, mas muitas vezes deixam de ouvir a voz do cliente em decisões importantes de marketing, vendas ou serviços. Isso pode resultar em produtos excessivamente sofisticados ou caros, que não atendem às necessidades reais do mercado.

As empresas orientadas à produção acreditam que os clientes escolhem produtos de baixo custo. Elas buscam manter os custos operacionais reduzidos e encontrar rotas de mercado mais econômicas. No entanto, essa abordagem pode não ser adequada para a maioria dos consumidores, que têm outras prioridades além do preço.

As empresas orientadas à venda partem do princípio de que, investindo suficientemente em publicidade, vendas e promoção, os clientes serão convencidos a comprar, independentemente de suas necessidades.

Por mais que muitos gestores defendam que a centralidade no cliente é a abordagem correta para todas as empresas, em diferentes estágios de desenvolvimento econômico ou de mercado, outras orientações podem ser mais atraentes. Toda empresa deve-se buscar a situação que se encontra e aplicar a melhor orientação (Buttle; Maklan, 2009).

- Operacional

O CRM operacional automatiza e melhora os processos de negócios de atendimento ao cliente e de suporte ao cliente. Os aplicativos de software de CRM permitem que as funções de marketing, vendas e serviços sejam automatizadas e integradas. As principais aplicações são os seguintes: automação de marketing, segmentação de marketing, gestão de campanhas, marketing baseado em eventos, automação de vendas, gestão contábil, gestão de *leads*, gestão de oportunidades, gestão de processos, gestão de contatos, geração de cotações e propostas, configuração de produtos, automação de serviços, gestão de incidentes e gestão de comunicações de entrada.

Para exemplificar uma aplicação desse tipo de CRM, o software de CRM deve possibilitar enviar disparos de campanhas aos seus consumidores para conseguir converter suas vendas. Os estudos desse projeto visa entender quais são os clientes mais propensos e com o software de disparo, conseguir enviar apenas para quem está mais propenso (Buttle; Maklan, 2009).

- Analítico

CRM Analítico é responsável por capturar, armazenar, extrair, integrar, processar, interpretar, distribuir, usar e reportar dados relacionados ao cliente para melhorar tanto o valor do cliente quanto da empresa. Esses dados podem ser encontrados em repositórios de vendas, finanças, marketing e serviços. Com aplicações de mineração, a empresa pode interrogar esses dados e responder perguntas para saber quem são seus consumidores mais valiosos, quais consumidores possuem maior propensão de mudar os competidores, quais consumidores são mais propensos a responder uma oferta particular, entre outros.

Esse tipo de CRM pode liderar as empresas a decidir abordagens diferentes entre os grupos de consumidores. Clientes com maior potencial é melhor vender presencialmente, clientes com menos potencial vender digitalmente. Além disso, o conteúdo e o estilo das comunicações podem ser feitos na medida, talvez por um segmento particular, usando análise do consumidor. Isso potencializa a probabilidade de uma oferta ser aceita pelo consumidor.

Do ponto de vista do consumidor, CRM pode entregar soluções customizáveis e pontuais para resolver os problemas do consumidor, aumentando assim sua satisfação. Do ponto de vista da empresa, podem prospectar maiores vendas cruzadas e aumento do número de vendas, melhorando a retenção e aquisição de consumidores (Buttle; Maklan, 2009).

- Colaborativo

CRM colaborativo é o termo utilizado para descrever o alinhamento tático e estratégico de separar empresas da cadeia de suprimentos para maior identificação, atração, retenção e desenvolvimento dos consumidores. Por exemplo, manufatureiros de bens do cliente e varejistas podem alinhar suas equipes, processos e tecnologias para servir os compradores mais eficientemente e efetivamente.

Desses tipos mencionados, esse projeto tem um foco muito maior no CRM Analítico pois com os dados dos históricos do consumidor, será usado um modelo estatístico para se retirar os consumidores estão mais propensos a realizar uma próxima compra (Buttle; Maklan, 2009).

Neste projeto, foi utilizado um software de CRM chamado Dito CRM. Com ele é possível ter uma visão boa do consumidor e é possível disparar campanhas de marketing com ele, haverá uma subseção para explicar como utilizá-lo. Mas basicamente, nesse software aplicam-se os tipos de CRM analítico e operacional. Compreendendo os conceitos básicos de CRM, será necessário entender como extrair os dados dos consumidores, saber quais dados são importantes para se realizar uma predição de propensão de compra, isso será visto na próxima seção de Machine Learning.

3.2 Machine Learning (Aprendizado de Máquinas)

3.2.1 Introdução ao Machine Learning

É a subárea da ciência da IA que dá habilidades ao computador para aprender sem ser explicitamente programado. Ela incorpora inúmeros algoritmos estatísticos e escolher o algoritmo correto ou a combinação de algoritmos para a tarefa é o grande desafio para quem trabalha nessa área. Há três tipos de categorias:

- Aprendizado Supervisionado

Essa categoria se baseia no aprendizado de padrões conectando as variáveis de entrada ou *features* em inglês e saída, ou *outcomes* em inglês. O aprendizado supervisionado funciona alimentando a máquina com várias *features* (representados como "X") e com o seu correspondente valor de saída (representado como "y"). O algoritmo decifra padrões que existem no dado e cria um modelo que consegue reproduzir as mesmas regras para um novo dado. Alguns exemplos de algoritmos dessa categoria são: regressão linear, árvores de decisão e redes neurais.

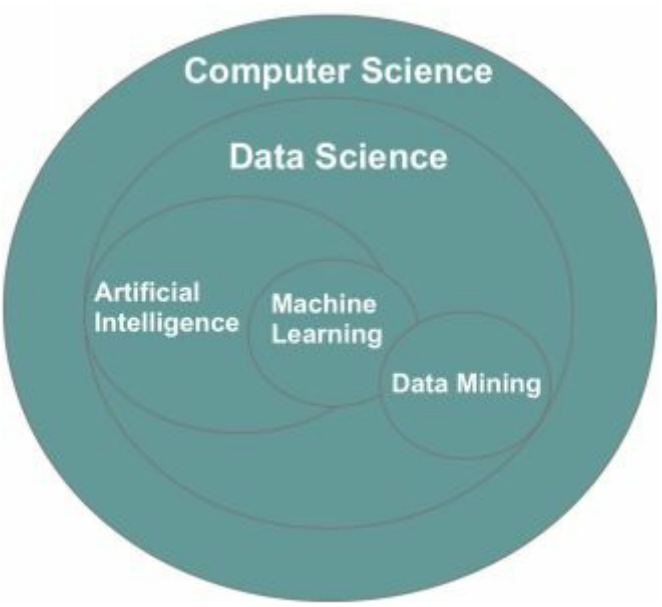
- Aprendizado não supervisionado

Nessa categoria não há classificações pré definidas. A máquina deve conseguir decifrar os valores das variáveis e a partir delas criar as classificações. A vantagem dessa categoria é poder descobrir padrões nos dados que um ser humano poderia não conseguir enxergar. Um grande exemplo de aplicação é na detecção de fraudes, o algoritmo deve ser capaz de analisar um comportamento anormal a partir de suas variáveis. Um exemplo de algoritmo é o K-Means.

- Aprendizado com reforço

O aprendizado com reforço continuamente melhora o modelo de aprendizado através de *feedbacks* com as interações anteriores. Um grande exemplo de aplicação é o computador aprender a jogar um jogo, cada partida o computador aprende a tomar decisões melhores, pois por decisões anteriores do jogo, o computador aprende o que deve-se fazer e não fazer em cada momento da partida. Um exemplo de algoritmo é o Q-learning (Theobald, 2017).

Figura 1 – Representação visual da relação entre as áreas de dados



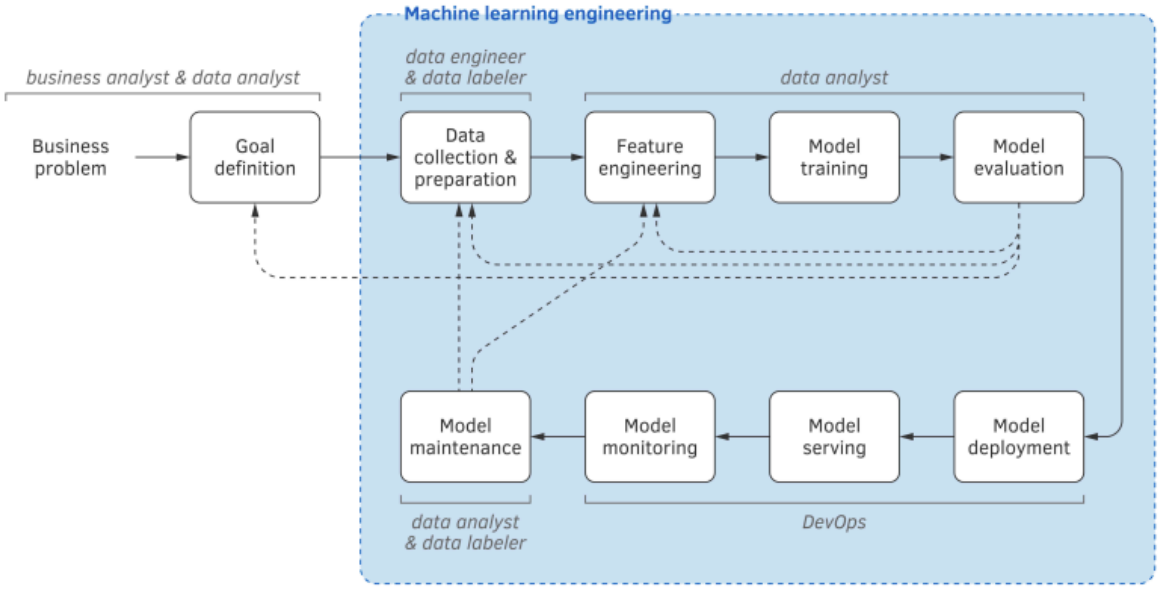
Fonte: Theobald (2017).

Um projeto de ML começa com um objetivo de negócio, o analista de negócios trabalha com o cliente e transforma o problema em um projeto de engenharia. O objetivo do aprendizado de máquina é construir um modelo que resolva o problema com um modelo que deve ser utilizado para automatizar, alertar, organizar, extrair, recomendar, classificar, etc. Se não for possível atingir o objetivo em uma dessas maneiras, ML não é a melhor solução a ser aplicada.

As razões para falhar um projeto de ML geralmente são falta de talento experiente, falta de suporte do líder, falta de infraestrutura de dados, desafio de rotular os dados, falta de alinhamento entre o time técnico e o time de negócios, entre outros. Portanto, antes de começar um projeto é muito importante saber que há inúmeras formas de fazer dar errado.

Em geral, o projeto tem nove etapas: definição do objetivo, coleção e preparação dos dados, engenharia dos atributos, treino do modelo, avaliação do modelo, desenvolvimento do modelo, serviço do modelo, monitoramento do modelo e manutenção do modelo. Neste projeto, terá enfoque nas etapas de coleção e preparação dos dados até o treinamento do modelo. A Figura 2 ilustra o ciclo de um desenvolvimento de um projeto de ML (Burkov, 2017).

Figura 2 – Ciclo de vida de um projeto de Machine Learning



Fonte: Burkov (2018).

3.2.2 Preparação e coleção dos dados

Antes de começar qualquer atividade de ML, é necessário preparar e coletar os dados. Os dados disponíveis nem sempre estão da maneira esperada. Nesta subseção serão apresentados propriedades de dados com boa qualidade, problemas que um *dataset* pode ter e maneiras de preparar e guardar dados para aplicações de ML Desse modo antes de começar a coletar os dados é importante verificar se o dado é:

- Acessível, verificando se respeita as normas, os direitos autorais.
- Considerável, no quesito de tamanho. É fundamental que possua dados suficientes para que possa treinar o modelo.
- Usável, não pode ter dados incompletos, absurdos ou duplicados, é necessário utilizar técnicas para resolver esses problemas.
- Entendível, não pode haver dados que são exatamente proporcionais ao valor de saída. Caso isso aconteça, o modelo irá rapidamente verificar essa similaridade e irá sempre retornar o valor correto, porém isso não é o que se almeja em uma aplicação de ML.
- Confiável, os dados não podem ter atrasos e nem serem indiretos, por exemplo, se há uma aplicação que almeja saber o que acontecerá daqui três semanas, então é necessário coletar dados que indiquem o estado para três semanas e não para a semana atual (Burkov, 2017).

Há alguns problemas comuns que acontecem os dados, sendo eles:

- Alto custo: Muitas vezes deve-se criar a própria maneira de coletar os dados de entrada, por exemplo, coletar dados de imagens, prover essas imagens e treinar esses modelos, podem ser muito custosos.

- Ruído: São os dados corruptíveis, por exemplo, em uma loja, o vendedor nem sempre pede os dados aos clientes, desse modo, os dados ficam incompletos e sem qual era o real dado. Há técnicas para solucionar esse problema, como a imputação de dados com operações estatísticas. Costuma-se ser um problema quando a base de dados é pequena, pois o ruído terá um maior efeito durante o treinamento do modelo, quando a base de dados é maior, há uma maior generalização e com isso o ruído terá um efeito menor.

- Viés: São as inconsistências que os dados representam Se faz a coleta dos dados em um meio particular e não generalizado, o modelo quando aprender, ele irá aprender as particularidades e dificilmente irá generalizar suas predições. Por exemplo, se na base há muitas avaliações positivas e poucas negativas para um produto, a tendência é sempre prever que as pessoas vão achar boas, o que é um problema para bons modelos.

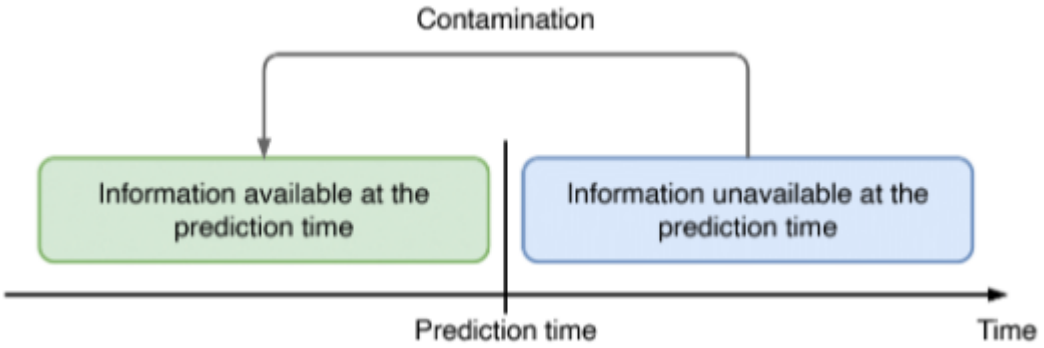
- Baixo poder preditivo: Muitas vezes os dados presentes não são o suficiente para ter um alto poder preditivo. Por exemplo, foi criado um modelo preditivo para saber se o ouvinte de uma playlist irá gostar da próxima música e utilizou apenas a letra da música, nome do artista e nome da música. Provavelmente terá muitos problemas, pois o ritmo e a frequência da música são fundamentais para o gosto musical também.

- Exemplos desatualizados: Assim que o modelo é desenvolvido e implantado, ela irá performar bem por um período. É importante que sempre treine os modelos uma vez que apresentarem um decaimento da performance. Esses decaimentos podem ser explicados por causa do conceito de desvio. As correlações entre as variáveis mudam de acordo com o tempo, elas não são fixas, por isso o modelo pode apresentar erros nas suas predições.

- Atipicidade: Dados atípicos são os dados que podem resultar em predições que não condizem com a realidade. Por exemplo, uma base de dados de compras, podem ter pessoas que compram cerca de 1 a 5 peças, porém nessa base há também as compras de revendedores. Os revendedores vão ter muito mais compras do que a maior parte da base. Mas o impacto desses dados podem ser ínfimos igual aos ruídos, uma vez que representam uma baixa proporção na base de dados, impactando pouco no treinamento do modelo e em sua generalização.

- Vazamento de dados: São os dados introduzidos involuntariamente sobre a variável de saída. Esse conceito pode ser chamado de contaminação, onde leva a performances muito boas, por exemplo, a variável de saída é a venda de uma casa e uma das variáveis de entrada é a comissão do vendedor. O modelo irá performar muito bem pois quando o exemplo apresentar comissão é porque foi vendido, acertando praticamente tudo. A imagem abaixo ilustra esse problema, a comissão é um evento posterior da predição, logo ela é prejudicial para o treinamento do modelo por esse motivo (Burkov, 2017).

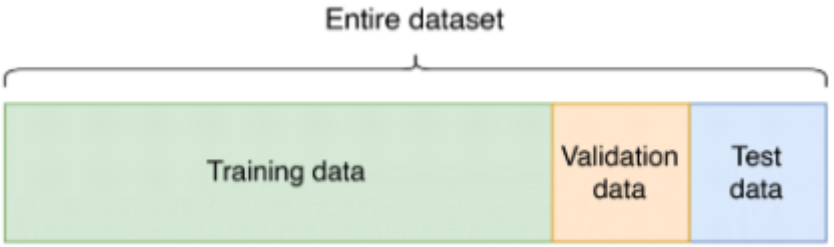
Figura 3 – Vazamento de dados de um modelo



Fonte: Burkov (2018).

Após saber os problemas comuns, uma boa base de dados possui as seguintes propriedades: contém informação suficiente que pode ser usado no modelo, tem uma boa cobertura do que se quer que faça no modelo, reflete a realidade, não possui viés, não é um resultado dela mesma, possui dados consistentes e é grande o suficiente para permitir generalizações.

Figura 4 – Particionamento da base de dados



Fonte: Burkov (2018).

A próxima etapa depois de construir a base de dados, é particioná-la em três partes: dados de treino, dados de validação e dados de teste. A base de treino, ou *training set* em inglês é usada para o algoritmo do aprendizado de máquina. A base de validação é usada para achar os melhores valores dos hiperparâmetros da *pipeline* do aprendizado de máquina. O analista deve tentar diferentes combinações de hiperparâmetros para conseguir os melhores valores, maximizando a performance do modelo. A base de teste é utilizada para verificar o resultado do modelo, conseguindo assim avaliar se o modelo está bom ou ruim.

Não há uma proporcionalidade bem definida na literatura, porém quando se há milhões de exemplos, organizações e cientistas dizem utilizar 70%/80% da base para treinar o modelo. O resto da base deverá ser dividido em partes iguais, logo 15%/15% ou 10%/10%.

Com os conceitos sobre a coleção e preparação dos dados, em seguida é necessário realizar a limpeza dos dados de entrada com técnicas de engenharia de ML, também conhecida como *feature engineering*, abordado na subseção a seguir (Burkov, 2017).

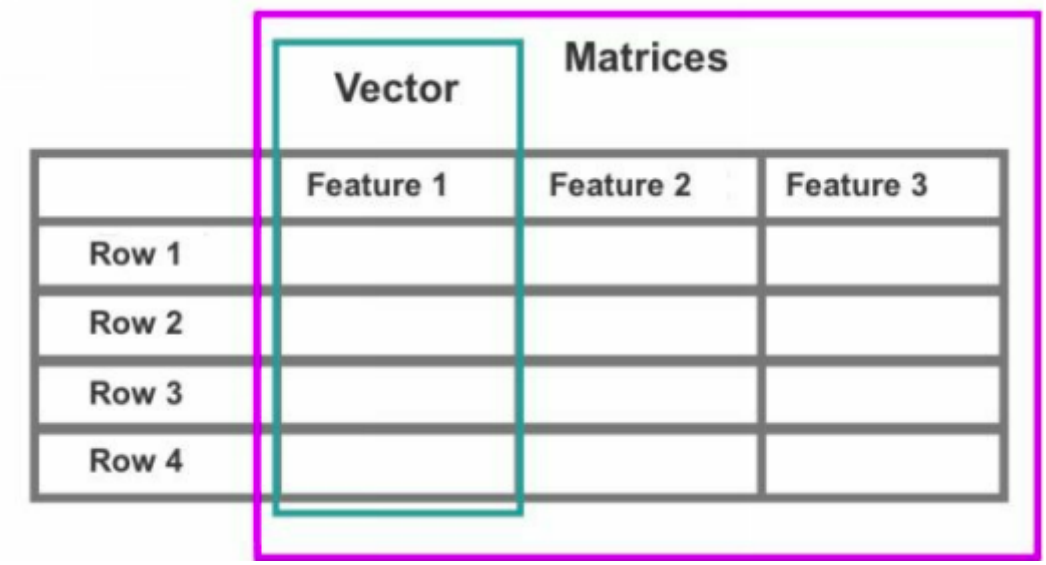
3.2.3 Feature engineering

Com os dados preparados e coletados, eles ainda vão estar “crus”, ou chamada de *raw* em inglês. É necessário transformar os dados em vetores de atributos que assim poderá ser possível realizar cálculos matriciais nos exemplos, esse processo é chamado de engenharia de atributos, ou *feature engineering* em inglês. A seguir será mostrado como os dados são organizados e quais são as principais transformações utilizadas na engenharia de atributos.

Os dados são constituídos por variáveis de entrada para formar a predição. O dado pode chegar de forma estruturada ou desestruturada. Os dados estruturados são dados categorizados em uma tabela definida por um esquema. Então esses dados são separados por linhas e colunas. Cada coluna é o que é chamado de *feature*, também conhecida como

variável (*variable* em inglês), ou atributo (*attribute* em inglês). Cada linha representa uma observação dada às *features*.

Figura 5 – Exemplo de um dataset tabular



Fonte: Theobald (2017).

Cada coluna é conhecida como vetor. Vetores armazenam os valores “X” (variáveis de entrada, ou *input*) e “y” (variável de saída, ou *output*) e o conjunto de vetores é chamado de matriz. Nota-se que nesse projeto será muito utilizado os termos em inglês para representar a entrada e saída. No caso do aprendizado supervisionado, o “y” é conhecido no dataset, enquanto no algoritmo não supervisionado não. Para exemplificar, o “y” pode ser o preço de um carro e o “X” todas as variáveis que compõem ele, como quilometragem (variável numérica), marca (variável classificatória), se há blindagem (variável booleana), entre outras. É necessário que a pessoa que irá treinar o modelo, tratar esses dados pois o algoritmo são equações e equações só trabalham com números (Theobald, 2017).

A seguir serão apresentados algumas técnicas de feature engineering:

- Seleção das *features*

Para gerar os melhores resultados dos dados é importante identificar as variáveis mais importantes. Utilizar variáveis que não são correlacionadas pode manipular e descarrilhar a acurácia do modelo. A Figura 6 a seguir possui um exemplo de como escolher as variáveis.

Figura 6 – Exemplo de um dataset tabular de línguas que podem ser descontinuadas

Name in English	Name in Spanish	Countries	Country Code
South Italian	Napolitano-calabres	Italy	ITA
Sicilian	Siciliano	Italy	ITA
Low Saxon	Bajo Sajón	Germany, Denmark, Netherlands, Poland, Russian Federation	DEU, DNK, NLD, POL, RUS
Belarusian	Bielorruso	Belarus, Latvia, Lithuania, Poland, Russian Federation, Ukraine	BRB, LVA, LTU, POL, RUS, UKR
Lombard	Lombardo	Italy, Switzerland	ITA, CHE

Fonte: Theobald (2017).

Nesse dataset, nota-se que duas primeiras colunas e as duas últimas representam o mesmo valor, desse modo é possível retirar a segunda coluna e a última por exemplo. Assim irá evitar complicações e potenciais imprecisões, melhorando também o processamento do modelo. Além da remoção de *features*, há a redução do número de *features* em menores. Na Figura 7 a seguir mostra um e-commerce de quatro compradores para 8 produtos.

Figura 7 – Exemplo de um dataset tabular de um e-commerce sem mescla

	Protein Shake	Nike Sneakers	Adidas Boots	Fitbit	Powerade	Protein Bar	Fitness Watch	Vitamins
Buyer 1	1	1	0	1	0	5	1	0
Buyer 2	0	0	0	0	0	0	0	1
Buyer 3	3	0	1	0	5	0	0	0
Buyer 4	1	1	0	0	10	1	0	0

Fonte: Theobald (2017).

As *features* poderiam ser reduzidas em tipos, mesclando variáveis semelhantes em menos colunas, separando em comidas saudáveis, vestiário e digital, como mostrado na Figra 8 a seguir.

Figura 8 – Exemplo de um dataset tabular de um e-commerce com mescla

	Health Food	Apparel	Digital
Buyer 1	6	1	2
Buyer 2	1	0	0
Buyer 3	8	1	0
Buyer 4	12	1	0

Fonte: Theobald (2017).

É muito importante saber entender que a separação deve-se ter sentido com o que se almeja. Por exemplo, se a aplicação for recomendar produtos para o comprador, essa separação faz sentido, mas caso a aplicação fosse saber qual marca o cliente mais gosta, não faria sentido juntar as marcas Adidas e Nike (Theobald, 2017).

- Compressão de linhas

É quase igual a redução do número de colunas, porém na visão de linhas. O ganho disso é no processamento e na redução de potenciais detrimentos à precisão do modelo (Theobald, 2017).

Figura 9 – Exemplo de um dataset tabular de uma corrida de animais

Before

Animal	Meat Eater	Legs	Tail	Race Time
Tiger	Yes	4	Yes	2:01 mins
Lion	Yes	4	Yes	2:05 mins
Tortoise	No	4	No	55:02 mins

After

Animal	Meat Eater	Legs	Tail	Race Time
Carnivore	Yes	4	Yes	2:03 mins
Tortoise	No	4	No	55:02 mins

Fonte: Theobald (2017).

- One-hot Encoding

Essa é a principal técnica para converter valores categóricos (Strings) em numéricos. Transforma o valor em binários (“1” e “0”). A tabela a seguir mostra um exemplo de dados que são categóricos.

Figura 10 – Exemplo de um dataset tabular de línguas que podem ser descontinuadas com o seu potencial de extinção antes da aplicação do one-hot encoding

Name in English	Speakers	Degree of Endangerment
South Italian	7500000	Vulnerable
Sicilian	5000000	Vulnerable
Low Saxon	4800000	Vulnerable
Belarusian	4000000	Vulnerable
Lombard	3500000	Definitely endangered
Romani	3500000	Definitely endangered
Yiddish	3000000	Definitely endangered
Gondi	2713790	Vulnerable
Picard	700000	Severely endangered

Fonte: Theobald (2017).

Figura 11 – Exemplo de um dataset tabular de línguas que podem ser descontinuadas com o seu potencial de extinção com aplicação do one-hot enconding

Name in English	Speakers	Vulnerable	Definitely Endangered	Severely Endangered
South Italian	7500000	1	0	0
Sicilian	5000000	1	0	0
Low Saxon	4800000	1	0	0
Belarusian	4000000	1	0	0
Lombard	3500000	0	1	0
Romani	3500000	0	1	0
Yiddish	3000000	0	1	0
Gondi	2713790	1	0	0
Picard	700000	0	0	1

Fonte: Theobald (2017).

Com as duas tabelas acima é mais intuitivo de entender o que aconteceu, basicamente transformou os dados categóricos em colunas com valores binários indicando se o estado é verdadeiro, representado por “1” ou falso, representado por “0”. Deve-se tomar muito cuidado com essa aplicação, pois se há muitas classificações, haverá a criação de muitas *features*, implicando em um potencial desastre no processamento (Theobald, 2017).

- Binning

Binning é a técnica para converter valores numéricos em binários. Nem todos os dados numéricos são relevantes nas aplicações. Por exemplo, em uma venda de apartamento, a quilometragem do espaço da garagem não é um dado relevante, apenas se há vaga ou não. Desse modo, basta aplicar a mudança da quilometragem para um campo binário (Theobald, 2017).

- Imputação de dados

Às vezes a base de dados vem sem valor em alguns atributos, muito comum com bases que foram feitas à mão. Há algumas formas resolver como: removendo os exemplos que estejam com valores faltantes do dataset quando há uma grande base, usando algum algoritmo para preencher esses dados ou usando a técnica de imputação de dados.

Quando o atributo é numérico, uma técnica é substituir o valor faltante pela média ou pela moda do valor desse atributo em relação à base inteira, outra é preenchendo com um valor que é fora da faixa dos dados, por exemplo, se os dados vão de 0 a 1, então preencher com -1. Quando o atributo é categórico, pode-se preencher com valor “Desconhecido”, o algoritmo vai reconhecer que é um valor diferente dos regulares (Burkov, 2017).

- Sobreamostragem ou subamostragem

Às vezes as bases de dados estão desbalanceadas, ou seja, há muita amostragem para um certo grupo e pouca amostragem de outro grupo. Isso pode desencadear problemas de enviesamento como visto anteriormente, então para resolver esse problema, pode-se ser utilizado a sobreamostragem (*oversampling)*, a qual são criados dados sintéticos para obter exemplos para a classe minoritária, enquanto a subamostragem (*undersampling*) remove dados da classe majoritária para se equilibrar com a minoritária.

Figura 12 – Sobreamostragem (esquerda) e subamostragem (direita)

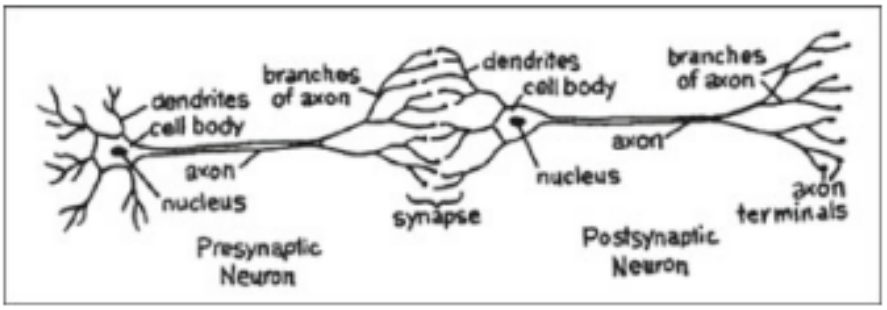
Fonte: Burkov (2018).

Há duas principais estratégias para se realizar uma amostragem. A primeira é a amostragem aleatória simples, no qual todos os exemplos podem ser escolhidos com a mesma chance, então basta aplicar alguma função de aleatoriedade e escolher o número desejado de features, ele pode não ser interessante quando os dados não estão englobando todos os grupos possíveis, isso acontece muito em CRM onde a maioria da base são leads e possuem uma característica particular nos atributos, podendo enviesar o treinamento do modelo. A segunda é a amostragem estratificada, na qual se sabe os grupos existentes na base e assim fazer uma amostragem aleatória para cada grupo, mantendo sua proporcionalidade, evitando assim que tenha enviesamento dos dados (Burkov, 2017).

3.2.4 Redes Neurais

Agora que foi abordado como gerar uma base sólida, deve-se saber como funciona o algoritmo. Redes neurais artificiais são técnicas de ML populares que simulam o mecanismo de aprendizado dos organismos biológicos. O sistema nervoso humano contém células que são referidas como neurônios. Os neurônios são conectados em outros pelo uso de axiomas e dendritos e a região que os conectam se chama de sinapses. A Figura 13 abaixo ilustra isso.

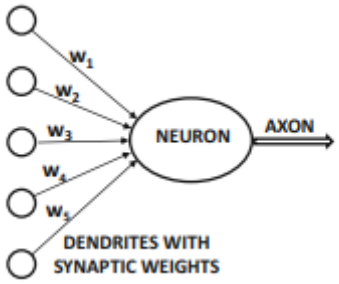
Figura 13 – Rede neural biológica



Fonte: Aggarwal (2023).

As forças das conexões sinápticas sempre mudam de acordo com o estímulo externo, essa mudança muda de acordo com a vivência dos organismos. Em termos computacionais, as unidades são conectadas entre as outras por meio de pesos, que possuem a mesma função das forças das conexões sinápticas. A Figura 14 abaixo ilustra essa representação computacional.

Figura 14 – Rede neural artificial



Fonte: Aggarwal (2023).

O aprendizado ocorre por causa das mudanças dos pesos que conectam os neurônios. Assim como os organismos biológicos precisam de estímulos externos, as redes neurais precisam de dados de treino, que contém exemplos de pares de *input-output* para a função que será treinada. Os dados de treino provisionam *feedbacks* para corrigir os pesos dependendo de quão bem a predição do *output* está condizente com um *input* particular. O objetivo de ajustar os pesos entre os neurônios é para modificar a função computada para realizar predições mais corretamente em futuras iterações. Os pesos são modificados cuidadosamente por um meio matemático para reduzir o erro computado. Se ajustar corretamente os pesos sucessivamente, a rede neural irá prover predições mais assertivas. Assim, o importante aprendizado sobre esse tema é saber como essas unidades combinadas dependendo da arquitetura podem realizar tarefas incríveis (Aggarwal, 2023).

- Perceptron

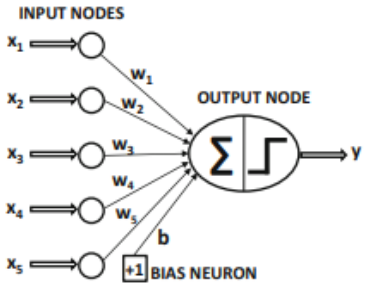
A rede neural mais simples é conhecida como perceptron. Essa rede neural contém uma única camada de entrada e um nó de saída, uma camada pode ter um ou vários nós. A arquitetura básica de um perceptron é apresentada na Figura 15. Os dados de treino que possuem d *inputs* e um único *output*, podem ser representados como um vetor de linha $\underline{X}$ e o *output* como y. A camada de *input* contém d nós que transmitem d *features*, assim tendo $\underline{X} = [x_1 \dots x_d]$. E a camada de *input* possui seus pesos, que assim como as *features* possui

um vetor de representação, também é representada por $\underline{w} = [w_1 \ldots w_d]^T$. A função linear $\underline{W} \cdot \underline{X}^T = \Sigma_{i=1}^{d} w_i x_i$ é computada no nó de *output*. Subsequentemente, um sinal desse valor é usado para predizer a variável dependente de $\underline{X}$. A predição $\hat{y}$ é computada como:

$$\hat{y} = f(\underline{X}) = sign\{\underline{W} \cdot \underline{X}^T\} = sign\{\Sigma_{j=1}^{d} w_j x_j\} \qquad (1)$$

A função de sinal mapeia o valor real entre +1 e -1, o qual é apropriado para classificação binária, mas não necessariamente precisa ser ela, mais para frente será abordado o uso dessa função, que generalizando se chama de função de ativação. A notação para indicar que é predição é com acento circunflexo $\hat{y}$ e o valor original é sem o acento y. A função de perda, ou de custo, a qual será abordada na subseção 3.2.5, é utilizada para verificar a diferença de erro entre o valor predito $\hat{y}$ e o valor y original. Os pesos da rede neural são treinados para minimizar a somatória dos erros sobre todas as instâncias de treino.

Figura 15 – Perceptron



Fonte: Aggarwal (2023).

Na figura 15 há a presença do viés, conhecido como *bias*. Muitas vezes os *inputs* e *outputs* não necessariamente possuem uma relação que passam pela origem, essas relações precisam de um deslocamento linear ajustando-as tanto para cima quanto para baixo. Então a equação 1 passa a ser mais representativa se for da seguinte maneira (Aggarwal, 2023):

$$\hat{y} = sign\{\underline{W} \cdot \underline{X}^T + b\} = sign\{\Sigma_{j=1}^{d} w_j x_j + b\} \qquad (2)$$

- Escolha da função de ativação

A escolha da função de ativação é uma parte crítica para o design da rede neural. Diferentes tipos de função não lineares como *sign* , *sigmoid*, *hyperbolic tangents* ou *ReLU*

são os mais comuns. A notação utilizada para representar a função é ϕ. Então, modificando a equação 1 com função de ativação genérica é denotada por:

$$\hat{y} = \phi\{\underline{W} \cdot \underline{X}^T\} \quad (3)$$

As funções não lineares mencionadas acima são representadas por:

$$\phi(v) = sign(v) \text{ (função } sign) \quad (4)$$

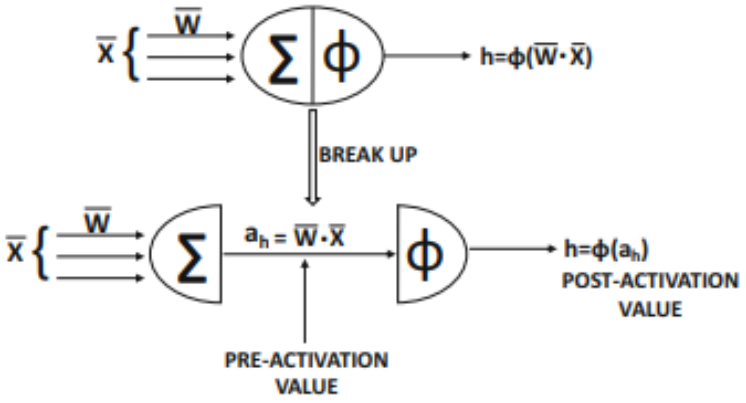
$$\phi(v) = \frac{1}{1+e^{-v}} \text{ (função } sigmoid) \quad (5)$$

$$\phi(v) = \frac{e^{2v}-1}{e^{2v}+1} \text{ (função } tanh) \quad (6)$$

$$\phi(v) = max\{v, 0\} \text{ (função } ReLU) \quad (7)$$

Para ilustrar o fluxo da matemática no neurônio, a figura 16 abaixo mostra a separação dos valores. O neurônio precisa computar tanto a função de somatória representado pelo símbolo Σ, quanto a função de ativação, representado pelo símbolo ϕ.

Figura 16 – Pré e pós ativação dos valores com o neurônio



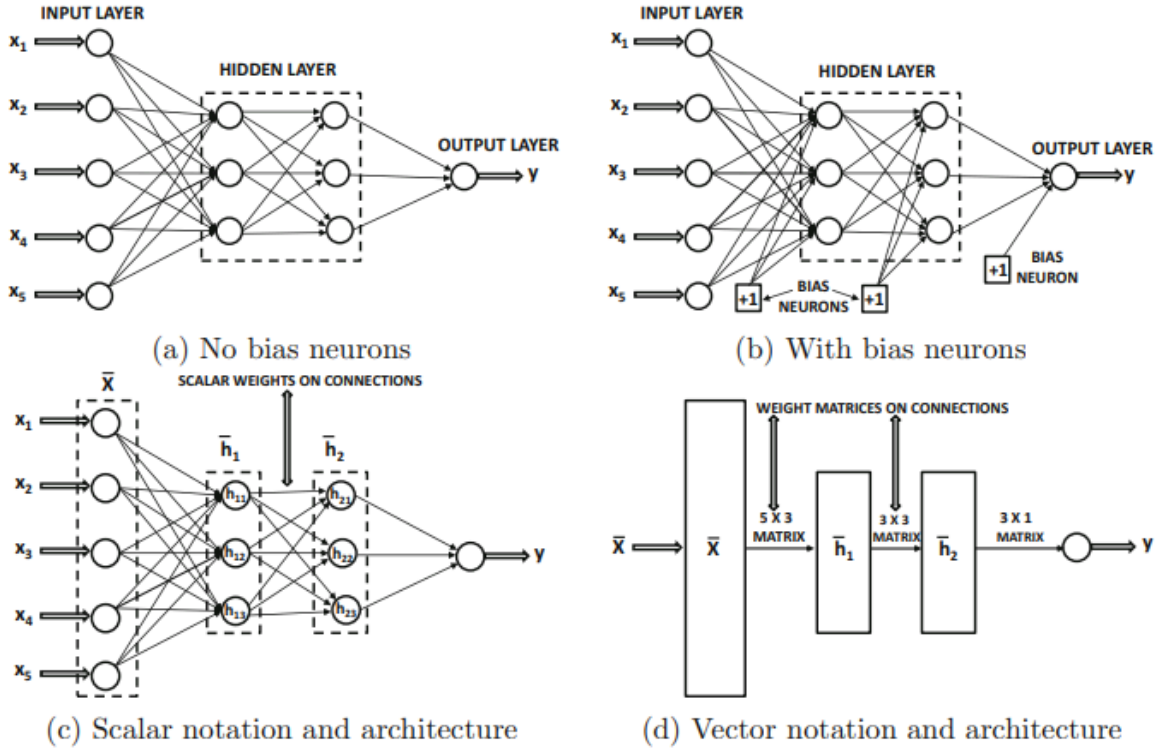
Fonte: Aggarwal (2023).

A importância de se utilizar a função de ativação é porque ela ajuda a criar composições poderosas de diferentes tipos de funções. O uso da não linearidade tem um papel fundamental O uso da não linearidade é fundamental para que o modelo consiga aprender e identificar padrões mais sofisticados, que não seriam detectáveis com modelos lineares (Aggarwal, 2023).

- Redes neurais multicamadas

Redes neurais multicamadas contém mais do que uma camada, camadas intermediárias são adicionadas e são denominadas *hidden layers*, porque suas computações não são visíveis ao usuário. A figura 17 abaixo representa uma rede neural de multicamadas.

Figura 17 – Arquitetura básica de uma rede feed-forward com duas camadas intermediárias e uma única camada de saída



Fonte: Aggarwal (2023).

Se uma rede neural possui $p_1..p_k$ unidades em cada k camadas, então o vetor representa esses *outputs* denotados por $\underline{h}_1 \ldots \underline{h}_k$ que tem dimensionalidade de $p_1..p_k$. Os pesos das conexões entre a camada de *input* e a primeira camada intermediária são contidas na matriz W_1 com o tamanho de $p_1 \times d$, enquanto os pesos entre a camada intermediária rº e a camada intermediária (r+1)º possuem o tamanho de $p_{r+1} \times p_r$ matriz, também chamada de W_r. Nota-se que o número de unidades da camada (r+1)º define o número de linhas enquanto o número de unidades da camada rº define o número de colunas. Se a camada de saída contém "o" nós, então a matriz final W_{k+1} tem o tamanho de $o \times p_k$ (Aggarwal, 2023).

- Retropropagação

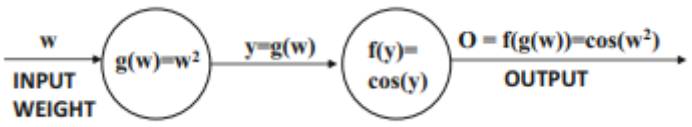
Entendendo como os neurônios geram uma predição através das multiplicações de matrizes entre seus valores e aplicando funções de ativações, é necessário entender como seus valores são atualizados para ter o mínimo de erro possível nas predições. A rede neural é uma computação gráfica, em que cada unidade de computação é um neurônio. Uma rede multicamadas avaliam composições de funções calculadas em cada nó individual. Um caminho de tamanho 2 em uma rede neural em que a função $f(\cdot)$ segue $g(\cdot)$ pode ser considerada uma função composta $f(g(\cdot))$. Em computação gráfica de dois nós e que é aplicada uma função sigmoide (equação 5) em cada nó que o peso é w, a equação pode ser dada por:

$$f(g(w)) = \frac{1}{1+exp[-\frac{1}{1+exp(-w)}]} \tag{8}$$

Retropropagação, ou *backpropagation,* é o método que se utiliza para obter o gradiente dos pesos em relação ao erro e assim é aplicado um outro método, conhecido como descida do gradiente, ou *gradient descent*, em que se atualizam esses pesos de acordo com o gradiente (Geeksforgeeks, 2025). Em um gráfico de curva simples de um vale, ou seja, há um ponto com valor mínimo, quando se calcula a derivada, ela indica a inclinação do ponto, para se encontrar o erro mínimo, deve-se achar o ponto em que a inclinação é 0. A mesma coisa deve acontecer em redes neurais, a ideia é ter o menor erro possível, logo ajustam-se os pesos de acordo com sua derivada para ter o menor erro possível nas predições.

Uma maneira de compreender *esse método* é abstraindo em computação gráfica, onde há nós e arestas para representar a rede neural, a figura 18 ilustra um exemplo disso.

Figura 18 – Simples computação gráfica com dois nós



Fonte: Aggarwal (2023).

Deve-se conhecer previamente a regra de cadeia de cálculos diferenciais, a equação 9 abaixo representa a versão mais simples:

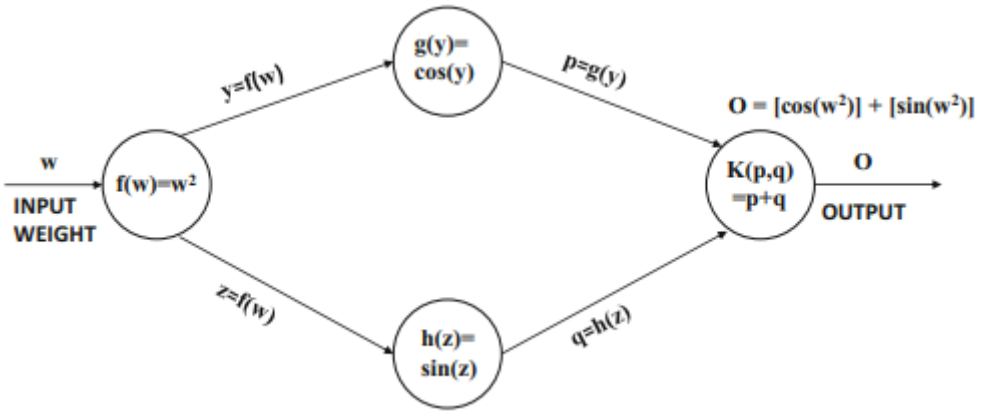
$$\frac{\partial f(g(w))}{\partial w} = \frac{\partial f(g(w))}{\partial g(w)} \cdot \frac{\partial g(w)}{\partial w} \tag{9}$$

Então, por exemplo, em uma rede simples com um neurônio apresentado na figura 18, para se realizar a derivada dessa computação gráfica, basta aplicar o conceito de cálculo derivacional da regra de cadeia, então o resultado se dá pela seguinte forma, expressado na equação abaixo:

$$\frac{\partial f(g(w))}{\partial w} = \frac{\partial f(g(w))}{\partial g(w)} \cdot \frac{\partial g(w)}{\partial w} = \; - \; sen(g(w)) \cdot 2w \; = \; - \; 2w \cdot sen(w^2) \qquad (10)$$

Mas como se sabe, em uma rede neural, dificilmente será um caminho simples como mostrado na figura 19, então adicionando um nó a mais na camada intermediária, obtém-se o seguinte exemplo:

Figura 19 – Simples computação gráfica com dois nós intermediários



Fonte: Aggarwal (2023).

O cálculo do gradiente de "o" para essa função é:

$$\frac{\partial o}{\partial w} = \frac{\partial K(p,q)}{\partial p} \cdot g'(y) \cdot f'(w) \; + \frac{\partial K(p,q)}{\partial p} \cdot h'(z) \cdot f'(w)$$

$$= 2w \cdot sen(y) \; + \; 2w \cdot cos(z)$$

$$= 2w \cdot sen(w^2) \; + \; 2w \cdot cos(w^2) \qquad (11)$$

E se for aplicar isso de uma forma geral, em que uma unidade recebe "k" nós, será necessário aplicar a regra de cadeia multivariável, expressa por:

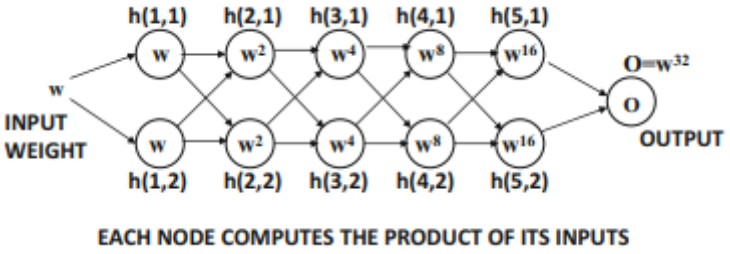
$$\frac{\partial f(g_1(w) \ldots g_k(w))}{\partial w} = \sum_{i=1}^{k} \frac{\partial f(g_1(w) \ldots g_k(w))}{\partial g_i(w)} \cdot \frac{\partial g_i(w)}{\partial w} \qquad (12)$$

Considerando em um grafo computacional em que o iº nó contém a variável $y(i)$, a sua derivada local $z(i,j)$ da sua aresta direta (i,j) no grafo será definida como $z(i,j) = \frac{\partial y(j)}{\partial y(i)}$. Haverá P caminhos existentes para a variável w no grafo para o nó de saída contendo a variável "o". Então o valor de $\frac{\partial o}{\partial w}$ é dado por calcular o produto de todos os gradientes locais de cada caminho em P e somando todos os valores desses produtos ao longo de todos os caminhos.

$$\frac{\partial o}{\partial w} = \sum_{P \in P} \prod_{(i,j) \in P} z(i,j) \tag{13}$$

Essas equações não são muito amigáveis de se entender, então sempre será mostrado com um exemplo visual para simplificar o entendimento, a figura 20 a seguir representa um grafo computacional de 5 camadas com 2 nós em cada uma.

Figura 20 – Computação gráfica com cinco nós intermediários



Fonte: Aggarwal (2023).

A jª unidade intermediária da iª camada é denotada como $h(i,j)$. Cada unidade intermediária é dada pelo produto de suas entradas, resultando em:

$$h(i,j) = h((i-1),1) \cdot h(i-1,2) \quad \forall j \in \{1,2\} \tag{14}$$

A derivada de cada $h(i,j)$ de acordo com suas duas entradas é expressa por:

$$\frac{\partial h(i,j)}{\partial h((i-1),1)} = h((i-1),2), \frac{\partial h(i,j)}{\partial h((i-1),1)} = h((i-1),1) \tag{15}$$

Vale ressaltar que são os resultados da equação 15 que são almejados, pois elas representam o gradiente de cada aresta, que no caso das redes neurais são os pesos. Então o valor do gradiente de "o" em relação aos pesos, dado por $\frac{\partial o}{\partial w}$, pode ser expresso por:

$$\frac{\partial o}{\partial w} = \sum_{j_1,j_2,j_3,j_4,j_5 \in \{1,2\}^5} \prod h(1,j_1)h(2,j_2)h(3,j_3)h(4,j_4)h(5,j_5)$$

$$= \sum_{Todos\ os\ 32\ caminhos} w^{31}$$

$$= 32w^{31} \tag{16}$$

Então, de forma resumida, existem essas duas fases com o aprendizado anterior:

- Fase de propagação: Nessa fase, um vetor de entrada particular é usado para calcular os valores de cada camada intermediária baseado nos valores atuais dos pesos. O objetivo dessa fase é calcular todas as variáveis de intermediárias e de saída dado uma entrada. O valor "o" é calculado, assim como o valor da perda L também é calculada.
- Fase de retropropagação: Nessa fase, calcula-se os valores dos gradientes das perdas em relação a vários pesos. O primeiro passo é calcular a derivada $\frac{\partial L}{\partial o}$. Subsequentemente, as derivadas são propagadas retroativamente utilizando a regra de cadeia multivariável. O caminho é denotado pela sequência das unidades intermediárias h_1, h_2, …, h_k seguidos pelo *output* "o". Os pesos das conexões das unidades intermediárias da unidade h_r até h_{r+1} é denotada por $w_{(h_r,h_{r+1})}$. Mas como se sabe, podem existir inúmeros caminhos de qualquer nó h_r até "o". Como visto na equação 13, a derivada parcial pode ser calculada agregados todos os produtos das derivadas parciais de todos os caminho de h_r até "o". A derivada da função de perda pode ser escrita como:

$$\frac{\partial L}{\partial w_{(h_{r-1},h_r)}} = \frac{\partial L}{\partial o} \cdot \left[\sum_{[h_r,h_{r+1},\ldots,h_k,o} \frac{\partial L}{\partial h_k} \prod_{i=r}^{k-1} \frac{\partial h_{i+1}}{\partial h_i} \right] \frac{\partial h_r}{\partial w_{(h_{r-1},h_r)}} \tag{17}$$

E se utilizar função de ativação, obtém que:

$$h_r = \phi(a_{h_r}) \tag{18}$$

Então a equação 17 ficará da seguinte forma:

$$\frac{\partial L}{\partial w_{(h_{r-1},h_r)}} = \frac{\partial L}{\partial o}\cdot\phi'(a_o)\cdot[\sum_{[h_r,h_{r+1},\ldots,h_k,o} \frac{\partial a_o}{\partial a_{h_k}} \prod_{i=r}^{k-1} \frac{\partial a_{h_{i+1}}}{\partial a_{h_i}}]h_{r-1} \quad (19)$$

Com a equação acima, será possível calcular o gradiente de cada peso da rede neural, possibilitando assim realizar sua atualização, com métodos matemáticos em que o mais conhecido é a descida do gradiente, ou *gradient descent* (Aggarwal, 2023).

- Descida do gradiente

Para minimizar o erro da função de perda da rede neural, basta movimentar os parâmetros no valor negativo correspondente ao gradiente. Por exemplo, em um perceptron, em que os pesos são definidos por $\underline{W} = (w_1 \ldots w_d)$. Então uma maneira de se atualizar é aplicando uma multiplicação do gradiente de acordo com o peso com um coeficiente α, chamado de taxa de aprendizagem, ou *learning rate*. A equação se dá por (Aggarwal, 2023):

$$\underline{W} \Leftarrow \underline{W} - \alpha(\frac{\partial L}{\partial w_1}, \frac{\partial L}{\partial w_2}, \ldots, \frac{\partial L}{\partial w_d}) \quad (20)$$

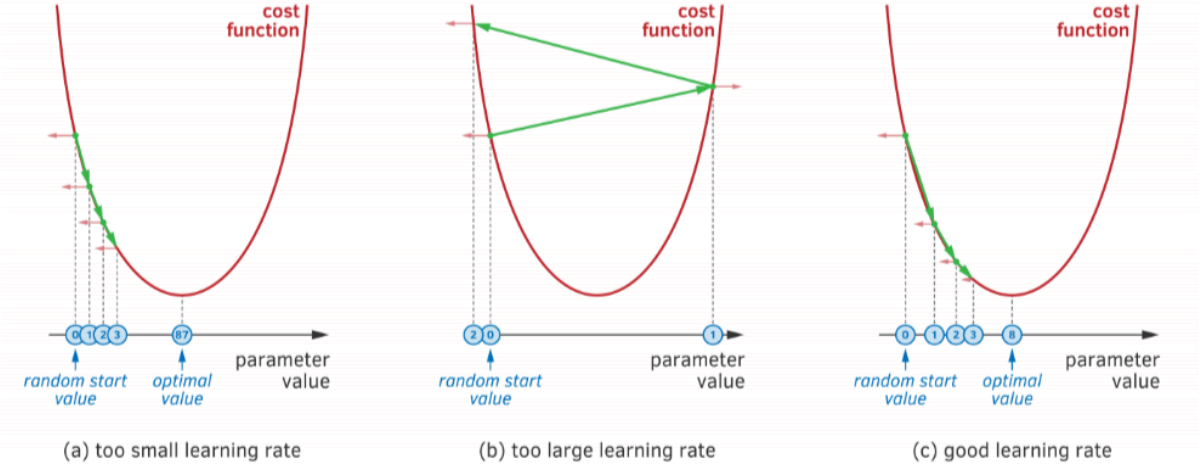
Gradient descent em ML procede em épocas (*epochs*). Uma época consiste em utilizar o conjunto de treino inteiro para atualizar cada parâmetro. Na primeira época, se inicializa com os parâmetros da rede neural usando uma das estratégias de inicialização que será abordado na subseção seguinte. A cada época, a descida de gradiente atualiza todos os parâmetros utilizando derivadas parciais. *Learning rate* controla a significância de uma atualização. O processo continua até a convergência, o estado quando os valores dos parâmetros não mudam muito após cada época e então o algoritmo para.

Escolher o *learning rate* correto para o problema não é fácil. Se escolher um valor muito alto, talvez não encontre a convergência. E do outro lado, se escolher um valor muito baixo, irá fazer com que o aprendizado demore. A figura abaixo retrata a ilustração da descida do gradiente para um parâmetro de uma rede neural. O valor de cada parâmetro em cada iteração é representado pelo círculo azul. O número dentro de cada círculo indica as épocas. A seta vermelha indica a direção do gradiente ao longo do eixo horizontal. A seta verde indica a mudança no valor da função de custo de cada época.

Em cada época, a descida do gradiente move o valor do parâmetro em direção ao custo mínimo. Essa otimização é lenta pois ela usa o conjunto de dados inteiro para computar

o gradiente de cada parâmetro em cada época. Há várias melhorias significativas para o algoritmo proposto. Uma delas é a descida do gradiente estocástica em mini-lotes em que se utilizam pequenos conjuntos de dados do conjunto de treino, aumentando a velocidade de processamento. Outra melhoria do algoritmo é ter uma taxa de aprendizado dinâmico com estatística com que fará encontrar o mínimo local mais eficientemente, os métodos mais famosos são Adam e Adagrad (Burkov, 2017).

Figura 21 – A influência da taxa de aprendizado para a convergência. (a) a taxa é muito baixa, a convergência será demorada; (b) é muito alta, não irá convergir; (c) a taxa está correta



Fonte: Burkov (2018).

Com o entendimento de como uma rede neural funciona, como seus pesos são atualizados, como suas predições são calculadas, é possível dar um passo a seguinte em que envolve todo o processo de treinamento do modelo.

3.2.5 Treinamento do modelo

Essa é a etapa mais simples de toda a engenharia, pois hoje temos muitas bibliotecas prontas que possuem todo o algoritmo por trás dos modelos, como Scikit-Learn e Tensorflow. A parte de coletar os dados e prepará-los para o treinamento é muito mais importante, pois o modelo irá refletir o que ele aprendeu, se ele estiver com uma base ruim, vai realizar predições ruins, se a base estiver boa, ele vai realizar predições boas. Antes de começar a

treinar o modelo, primeiramente deve-se verificar a conformidade do esquema, ou *schema*, dos dados, muitas vezes as fontes fazem mudanças que fazem o modelo falhar.

- Definição do nível de performance atingível

Deve-se definir um nível de performance atingível, ou seja, quando deve-se parar de procurar tentar melhorar o modelo. Caso um humano consiga fazer predições sem muitos esforços, sem matemática, sem derivações lógicas complexas, então é esperado que o modelo obtenha o mesmo nível humano de performance. Caso toda informação para realizar predições esteja contida nas variáveis, então é esperado ter um erro quase nulo. Se o vetor das variáveis estiver com um alto número de sinais (como de imagens), é esperado um erro quase nulo também. Se houver um programa que faz classificação ou regressão, é esperado que o modelo tenha performance similar ao programa e o modelo pode superar caso tenha bastante dados para treinar (Burkov, 2017).

- Escolha da função de custo

Deve-se escolher uma função de custo, pois ela será fundamental para direcionar o treinamento do modelo. Ela que mede o erro do modelo entre os valores reais e os valores preditos do conjunto de dados. Há um objetivo no treinamento do modelo que é reduzi-lo ao máximo. Para regressões é comum utilizar *mean squared error (MSE)* ou *median absolute error (MAE)*.

A Equação 21 abaixo mostra como se calcula o MSE de uma regressão:

$$MSE(f) = \frac{1}{N} \sum_{i=1...N} (f(x_i) - y_i)^2 \tag{21}$$

Nessa equação "x" é o vetor dos atributos (entrada) e "y" é a predição (saída) e "i" é um range de 1 a N que indica os exemplos do *dataset*. Se há presença de atipicidades, é melhor utilizar o MAE, uma vez que as atipicidades podem alterar drasticamente o valor do MSE. Sua equação é representada por:

$$MAE = mediana(\{|f(x_i) - y_i|\}_{i=1}^{N}), \tag{22}$$

onde $\{|f(x_i) - y_i|\}_{i=1}^{N}$ denota o conjunto de erros absolutos de todos os exemplos de 1 a N.

Para classificações binárias é comum utilizar a entropia cruzada categórica binária, ou *binary categorical cross-entropy* em inglês. A fórmula disso pode ser dada por:

$$BCE_i = -y_i \times log_2(\hat{y}_i) - (1 - y) \times log_2(1 - \hat{y}_i) \quad , \tag{23}$$

onde, $\hat{y}_i$ é o vetor de predição do modelo para o input x_i e y_i é o vetor de valor real de saída. A função de custo para classificações pode ser definida como a soma de todas as perdas, ou *loss* em inglês, de cada exemplo individual, resultando na seguinte equação:

$$BCE = \sum_{i=1}^{N} BCE_i \tag{24}$$

Quando realizar o treinamento, conforme for modificando os pesos, deve-se ter um valor mínimo de diferença entre o valor das perdas nas iterações para determinar a parada do treinamento (Burkov, 2017).

- Escolha da métrica de performance

Deve-se escolher uma métrica de performance, é com ela que se comparar diferentes modelos e verificar se é necessário adicionar *features* ou aumentar a complexidade do modelo. Para regressões é comum utilizar o R^2 (Coeficiente de Determinação) que é uma métrica estatística para medir a proporção da variância total da variável dependente. A Equação dela é determinada por:

$$R^2 = 1 - \frac{SSE}{SST} \tag{25}$$

onde SSE é a soma dos quadrados dos resíduos, ou *sum of squared errors* em inglês e SST é a soma total dos quadrados, ou *total sum of squares)*. O SSE representa a diferença entre os valores observados (y_i) e os valores previstos ($\hat{y}_i$) e o SST representa a variância total dos dados, considerando a média a $\underline{y}$ como predição de referência. As equações de SSE e SST são apresentados abaixo, respectivamente:

$$SSE = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{26}$$

$$SST = \sum_{i=1}^{N} (y_i - \underline{y}_i)^2 \tag{27}$$

Se $R^2 = 0$, então o modelo não é melhor que a média, se $R^2 = 1$, o modelo está ajustado perfeitamente. Então deve-se buscar maximizar o valor de R^2.

Para as classificações, as métricas mais utilizadas são: precisão-recall (*precision-recall*), acurácia (*accuracy*) e área sob a curva ROC (AUC). Essas métricas são facilmente calculadas quando se realiza a matriz de confusão (*confusion matrix*) antes, ela é uma tabela que resume o sucesso da classificação do modelo com suas predições e seus respectivos valores reais. Um exemplo abaixo indica uma predição de classificação, apresentado na Tabela 1:

Tabela 1 – Exemplo de matriz de confusão

	[Verdadeiro] predito	[Falso] predito
[Verdadeiro] real	20 (TP)	1 (FN)
[Falso] real	10 (FP)	500 (TN)

Fonte: Autoria própria.

De 21 casos que são verdadeiros, ou seja, pode ser uma indicação se é spam, uma pessoa vai realizar compra ou qualquer ação, o modelo classificou corretamente 20 casos (Verdadeiro Positivo/True Positive/TP) e um caso o modelo predisse incorretamente (Falso Negativo/False Negative/FN). E de 510 casos que são falsos, ou seja, o contrário da ação analisada, o modelo classificou 500 corretamente (Verdadeiro Negativo/True Negative/TN) e 10 incorretamente (Falso Positivo/False Positive/FP).

Com isso, a precisão é definida pela razão entre as predições positivas sobre todas as predições positivas, dada pela equação abaixo:

$$precision = \frac{TP}{TP + FP}, \tag{28}$$

recall é a razão entre as predições positivas e o número de exemplos positivos, dada pela equação abaixo:

$$recall = \frac{TP}{TP + FN} \tag{29}$$

Para entender melhor essas duas métricas, suponha um exemplo de detecção de spam em emails. A precisão responde a seguinte pergunta: “De todos os emails que o modelo classificou como spam, quantos realmente são spam?”, enquanto o recall busca responder a seguinte pergunta: “De todos os spam reais, quantos o modelo identificou corretamente?”.

Assim, é importante saber qual das duas métricas irá definir as metas, pois uma alta precisão é importante quando os falsos positivos são custosos, por exemplo, em classificações de email, marcar como spam um e-mail que não é, é muito prejudicial uma vez que o usuário poderá perder informações importantes. Um recall alto é importante quando os positivos verdadeiros faltantes são custosos, por exemplo, se detectar todos os spams são bons para segurança, então é importante que consiga detectar o maior número de spams possível. Porém nota-se que sempre um irá piorar o valor do outro, conhecido como *precision-recall tradeoff*.

Como existe essa troca, os modelos podem ter precisão maior e recall menor e vice-versa, assim para conseguir determinar qual é o melhor modelo, uma fórmula é usada que combina as duas métricas, chamada de F-score. A sua fórmula é definida pela equação abaixo:

$$F_1 = \left(\frac{2}{recall^{-1} + precision^{-1}}\right) = 2 \times \frac{precision \times recall}{precision + recall}, \quad (30)$$

e genericamente, utiliza-se um valor positivo real β para indicar que o recall é β vezes mais importante que a precisão, então a equação do F-score fica da seguinte forma:

$$F_\beta = (1 + \beta^2) \times \frac{precision \times recall}{(\beta^2 \times precision) + recall} \quad (31)$$

A acurácia é o número de classificações corretas pelo número total de exemplos classificados. Ela pode ser definida pela equação abaixo:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (32)$$

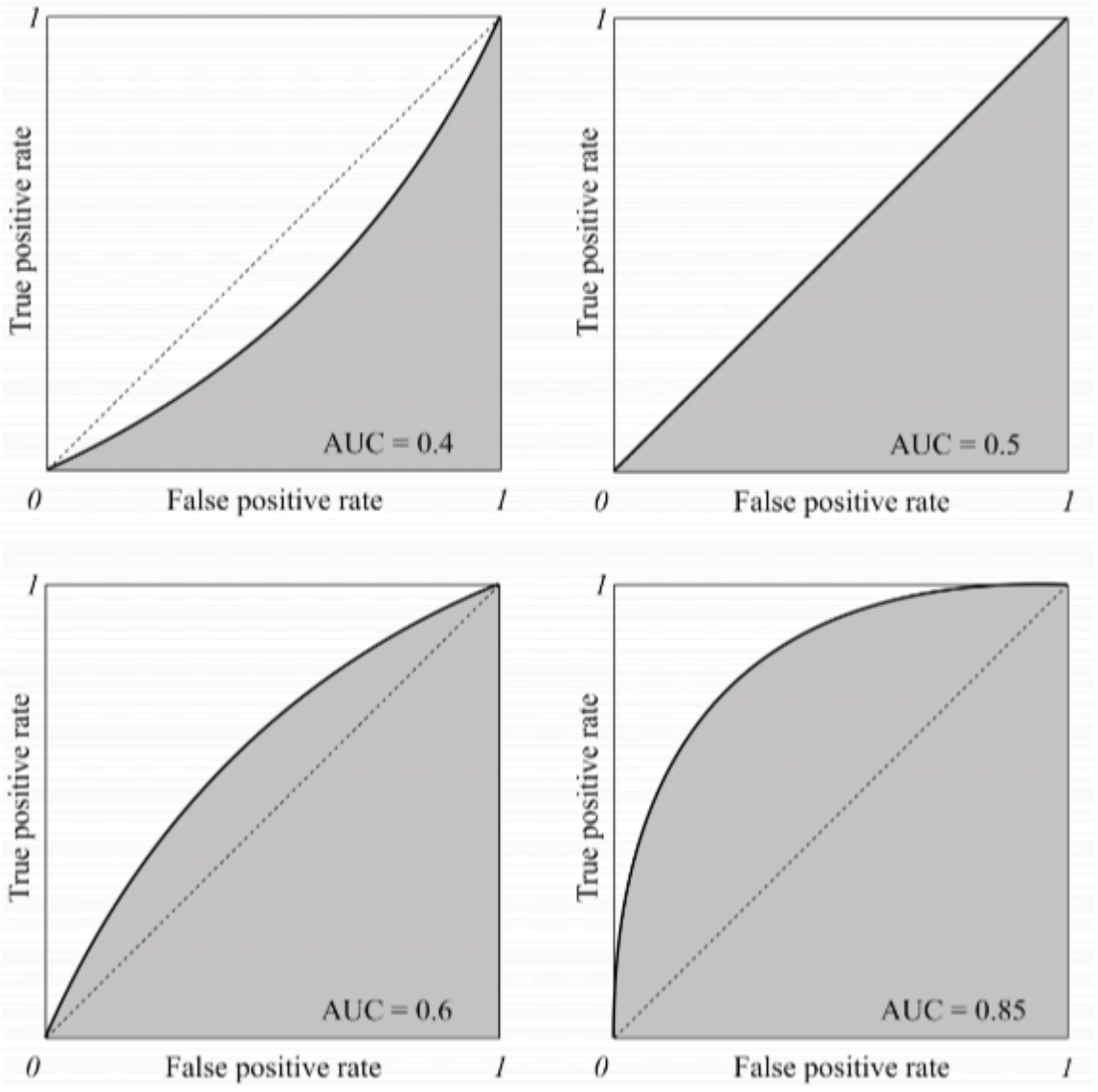
Essa métrica é importante quando os erros das predições são igualmente importantes, tanto da parte positiva quanto negativa. Com dados desbalanceados, essa é uma péssima métrica a ser utilizada, pois no exemplo de spam, a maioria dos e-mails não são spams, então se o modelo simplesmente dizer que tudo não é spam, irá ter uma boa acurácia, o que pode não indicar que o modelo está bom.

E por fim, a última métrica a ser detalhada é a *ROC (receiver operating characteristic) curve*. Essa métrica usa a combinação da razão de positivos verdadeiros, ou *true positive rate* e pela razão de positivos falsos, ou *false positive rate*. O *true positive rate* (TPR) e o *false positive rate* (FPR) podem ser definidas respectivamente como:

$$TPR = \frac{TP}{TP + FN} \; e \; FPR = \frac{FP}{FP + TN} \quad (33)$$

Para desenhar a *ROC curve*, deve-se criar um range de 0 a 1, como [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1], esses valores vão ser o limiar, ou *threshold*, das predições do modelo. Se o limiar é de 0,7 e a probabilidade de ser positivo é de 0,6, então o modelo irá predizer como negativo pois 0,6 está abaixo do *threshold.* Assim calcula-se todos os TPR e FPR para cada *threshold* e plota-se um gráfico onde o TPR fica no eixo Y e o FPR fica no eixo X. Quanto maior a área sob a *ROC curve*, ou *area under the ROC curve* (AUC), melhor é a classificação. Se o modelo está com AUC menor que 0,5 então há alguma coisa de errado e se é maior é que o modelo classifica melhor do que um modelo que classifica aleatoriamente. A figura 22 abaixo, indica um exemplo dessa curva.

Figura 22 – Área abaixo da curva ROC (AUC) mostrado em cinza



Fonte: Burkov (2018).

- Escolha da estratégia de inicialização dos parâmetros

Antes de começar o treino do modelo todos os valores dos parâmetros são desconhecidos. Deve-se começar iniciando com alguns valores. Há as estratégias abaixo:

- uns: todos os parâmetros são inicializados com 1.
- zeros: todos os parâmetros são inicializados com 0.
- normal aleatória (*random normal*): parâmetros são inicializados com valores amostrados pela distribuição normal, normalmente com a média de 0 e desvio padrão de 0,05.
- uniforme aleatória (*random uniform*): parâmetros são inicializados com os valores amostrados da distribuição uniforme com o range de [-0,05;0,05].
- normal de Xavier (*Xavier normal)*: parâmetros são inicializados com os valores amostrados da distribuição normal truncada, centralizada em 0, com desvio padrão igual a $\sqrt{2/(in + out)}$ onde, "in" é o número de neurônios na camada de entrada e de saída.

- Algoritmo de otimização

Basta aplicar um dos algoritmos que existem visto na subseção anterior.

- Ajuste de hiperparâmetros

Hiperparâmetros possuem um papel fundamental no processo de treinamento do modelo. Alguns influenciam na velocidade do treino, mas suas principais características são controlar as duas trocas: viés-variância (*bias-variance*) e precisão-recall (*precision-recall*). Os hiperparâmetros são determinados fazendo experimentos com combinações de valores, um para cada hiperparâmetro.

Há algumas técnicas de ajustes de hiperparâmetros, como:

- *Grid Search*: Essa técnica consiste em discretizar cada um dos hiperparâmetros e avaliar cada par dos valores discretos. Com as múltiplas combinações, basta escolher o melhor par com a métrica que se almeja ter melhores resultados.

- *Random Search*: Como o *grid search* pode ser muito custoso, pois é necessário realizar muitas combinações, o *random search* não necessita provisionar um conjunto de valores para explorar cada hiperparâmetro. Essa técnica provisiona uma

distribuição estatística para cada hiperparâmetro que cada valor é aleatoriamente amostrado.

- *Vizier:* É uma ferramenta do Google que implementa uma técnica avançada de ajuste de hiperparâmetros chamada Otimização Bayesiana. Ao contrário de abordagens como o *Grid Search* e o *Random Search*, que não levam em consideração os resultados das avaliações anteriores, a Otimização Bayesiana usa os resultados das combinações testadas para guiar a busca pelos melhores hiperparâmetros. O *Vizier* ajusta os parâmetros de forma inteligente, priorizando combinações que têm maior probabilidade de melhorar o desempenho do modelo, com base em um modelo probabilístico. Esse processo é iterativo e tende a encontrar melhores combinações de hiperparâmetros com menos experimentos, tornando-o mais eficiente em problemas com muitas variáveis e com alta complexidade.

- Regularização

Antes de entender a regularização é importante entender a troca de viés-variância (*bias-viariance tradeoff*). Nesse *tradeoff* há dois casos que são prejudiciais ao modelo:

- *Underfitting*: Quando o modelo está com pouco enviesamento (*low bias*), o modelo acerta muitas predições, quando o modelo está com muito enviesamento (*high bias*), o modelo costuma errar muitas predições, o qual é o caso desse problema. Alguns modelos apresentam *underfitting* por algumas razões: o modelo está muito simples para os dados, as variáveis de entrada não são informativas para realizar a predição ou houve muita regularização no modelo. Para resolver esse problema costuma-se criar modelos mais complexos, melhorar a engenharia dos atributos para ter um poder preditivo maior, adicionando mais dados de treino ou reduzindo a regularização.
- *Overfitting*: Nesse problema é quando o modelo acerta muitas predições porém quando testa com uma base genérica, ela tem uma má performance. Outro nome para *overfitting* é alta variância, ou *high variance*. Algumas razões fazem acontecer esse problema: o modelo está muito complexo para os dados, há muitos atributos e poucos exemplos de treino ou não houve regularização suficiente. Para resolver esse problema costuma-se usar um modelo mais simples, reduzir a dimensão dos exemplos do *dataset*, adicionar mais dados de treino ou regularizar o modelo.

Regularização é um método para forçar o algoritmo de aprendizado a treinar um modelo menos complexo. Os dois tipos mais utilizados são regularizações L1 e L2. São constantes que penalizam o algoritmo de aprendizado. A equação de L1 pode ser representada como:

$$min_{w^{(1)},w^{(2)},b}[C\times(|w^{(1)}| + |w^{(2)}|) + \frac{1}{N}\times\sum_{i=1}^{N}(f_i - y_i)^2], \quad (36)$$

e L2 pode ser representada como:

$$min_{w^{(1)},w^{(2)},b}[C\times((w^{(1)})^2 + (w^{(2)})^2) + \frac{1}{N}\times\sum_{i=1}^{N}(f_i - y_i)^2] \quad (37)$$

Nota-se que se C é 0, o modelo terá o seu estado normal para adquirir o erro do modelo, agora se colocar o C com um valor alto, o modelo irá deixar os pesos *w* com um valor próximo de zero para minimizar o erro. A diferença entre L1 e L2 está em suas dimensões, em que L2 os pesos estão na base elevada a dois.

3.2.6 Passo a passo do treinamento

Para se criar um modelo do zero, a estratégia segue esses nove passos:

1. Definir a métrica P
2. Definir a função de custo C
3. Escolher uma estratégia de parâmetro de inicialização W
4. Escolher um algoritmo de otimização de função de custo A
5. Escolher uma estratégia de ajuste de hiperparâmetro T
6. Escolher uma combinação H dos valores dos hiperparâmetros usando a estratégia T
7. Treinar o modelo M, usando o algoritmo A, parametrizado com os hiperparâmetro H, para otimizar a função de custo C
8. Se ainda houver hiperparâmetros não testados, escolher a combinação H dos valores dos hiperparâmetros utilizando a estratégia T e repetir o passo anterior.
9. Retornar o modelo com a métrica P otimizada

Com o modelo treinado, será possível colocar em prática todo o aprendizado da máquina.

3.3 BigQuery

Esta subseção será baseada no livro “Google BigQuery: O Guia Definitivo: Data Warehousing, Analytics e Machine Learning na Escala“. BigQuery é um serviço de armazenamento de dados sem servidor, altamente escalável que vem com um motor de consultas pré estabelecido, ou *built-in query engine* em inglês. *Query engine* do BigQuery é capaz de rodar terabytes de dados em segundos e petabytes de dados em minutos. É possível atingir essa performance sem precisar manusear a infraestrutura sem precisar criar ou recriar index. Com o avanço da cultura orientada a dados nas empresas, BigQuery está sendo uma importante ferramenta para essa inovação (Lakshmanan, 2019).

3.3.1 Introdução ao BigQuery

Agora serão abordadas as principais vantagens de se utilizar o BigQuery em relação aos outros bancos relacionais. Quando se escreve transações, provavelmente será escrito em base de dados de processamento de transações online relacionais (OLTP), como PostgreSQL e MySQL. Esses bancos são ruins quando se precisa realizar agregações, agrupamentos e ordenações, pois o foco deles é em grandes volumes de transações em tempo real, o foco dele é garantir a rapidez, consistência e confiabilidade nas operações. Devido a esse problema de conseguir processar inúmeros dados, foi implementado a framework MapReduce, a qual é uma abstração que possibilita computar em termos de dois passos: uma função para mapear (*map*) o par chave/valor processado para gerar um conjunto de chave/valor intermediário, e uma função para reduzir (*reduce*) que junta todos os valores intermediários associados com a mesma chave intermediária. Esse framework necessita de grupos (*cluster*) grande necessário para conseguir processar os dados e isso implica em um problema em que sempre fica ou super provisionado ou sobre provisionado, necessitando de especialistas em gerenciar, monitorar e provisionar *clusters*. O BigQuery é interessante por causa disso, é um serviço sem servidor que pode rodar consultas (*queries*) sem precisar gerenciar a infraestrutura, ele mesmo realizará o planejamento e a otimização das *queries*. As *queries* são automaticamente escaladas para milhares de máquinas e executadas em paralelo, não é necessário realizar nada especial para poder ativar a paralelização.

O BigQuery utiliza uma arquitetura de alto nível, a *query* é recebida por um servidor Google Front End (GFE) e roteado para o BigQuery Job Server apropriado e tratado pelo servidor de consultas Dremel. Os estágios dos diferentes tipos de consultas das mais simples como a consulta de varredura-filtragem-contagem (*scan-filter-count query*) para as mais

complexas, como a consulta de varredura-filtragem-agregação, até para com altas cardinalidades que exigem repartições.

Além da arquitetura, é importante saber como os dados são armazenados. A vantagem do BigQuery é que ela é em forma colunar, possui codificação por dicionário e o uso de conjunto de armazenamento que possibilita acessar dados de versões históricas. E particionamento e clusterização quando implementados, aumentam a performance das *queries* (Lakshmanan, 2019).

3.3.2 Organização dos recursos

- Datasets

Datasets são containers lógicos que são usados para controlar e organizar acessos dos recursos do BigQuery. *Datasets* são similares aos esquemas (*schemas*) em outros sistemas de banco de dados. A maioria dos recursos que são criados, incluindo tabelas, *views* e *procedures* são criados dentro de um *dataset*. Conexões e tarefas (*jobs*) são exceções, esses são associados com o projeto e não o *dataset* (Lakshmanan, 2019).

- Projetos

Todo *dataset* está associado a um projeto. Para usar o Google Cloud, é necessário criar ao menos um projeto. Projetos são a base para criar, permitir e usar todos os serviços da Google. Um projeto consegue possuir múltiplos *datasets*, e *datasets* de diferentes localizações pode existir no mesmo projeto.

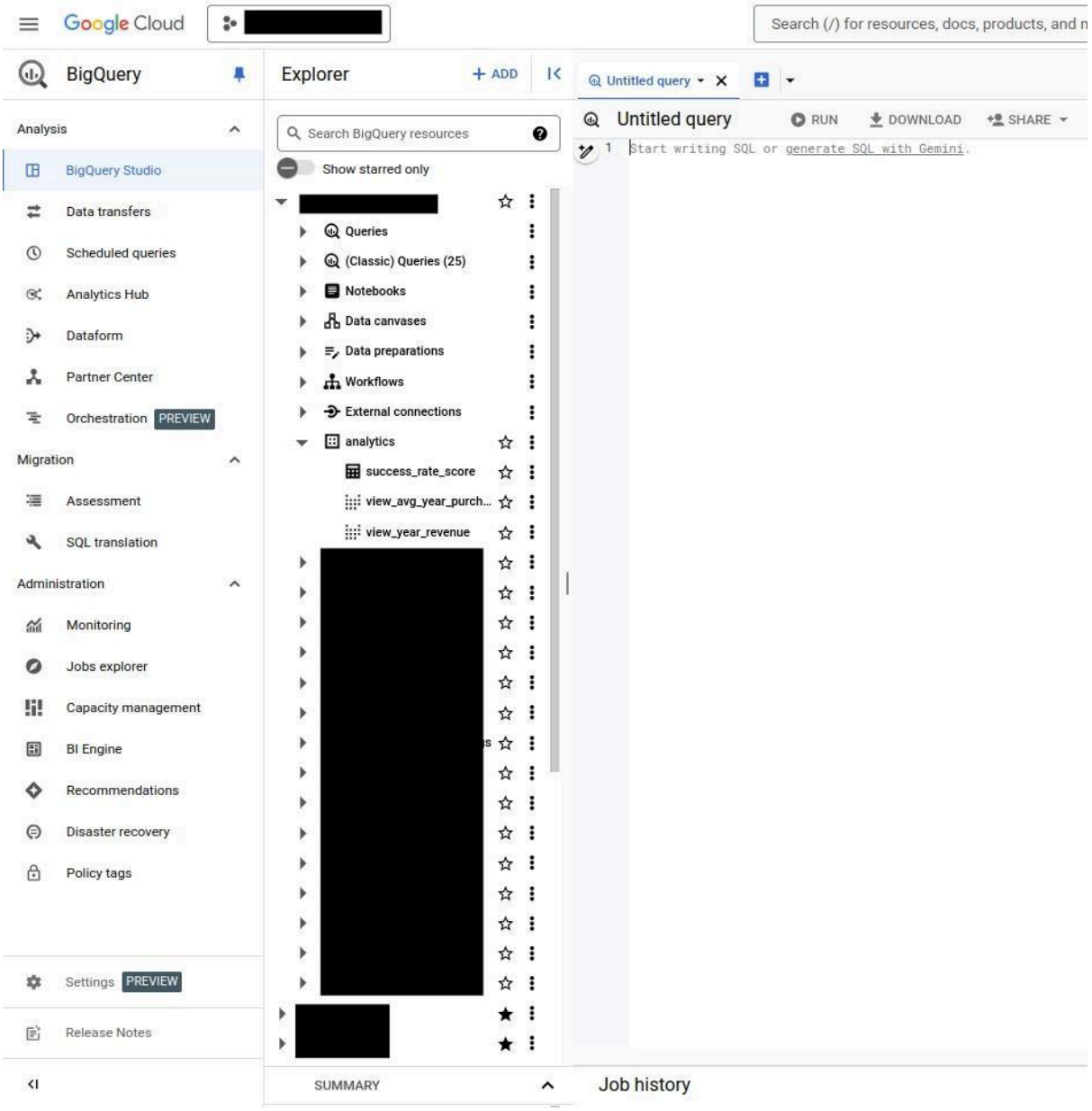
Quando se realiza operações nos dados do BigQuery, como rodar uma *query* ou realizar ingestão de dados em uma tabela, é criado um *job*. Um *job* é sempre associado com o projeto, mas não é necessário rodar no mesmo projeto que contém os dados e ele pode referenciar tabelas de *datasets* de múltiplos projetos (Lakshmanan, 2019).

- BigQuery Console

O Google Cloud provisiona uma interface gráfica que possibilita criar e gerenciar os recursos do Bigquery e rodar SQL *queries*. A figura 23 retrata a página do console. Na esquerda se encontra o menu de navegação, em que estão focados nas ferramentas para realizar extrações, transformações e cargas. Há a parte de administração em que é focado mais no monitoramento e gerenciamento de recursos. No meio, em “Explorer”, encontram-se os bancos de dados separados por projetos e níveis. O primeiro nível hachurado é o projeto, seguido das *queries* salvas, códigos em *notebooks*, canvas dos dados, preparação dos dados e conexões e por fim os *datasets* e tabelas e *views* atreladas a ele, por exemplo nesse caso, o *dataset* encontrado é o “analytics” e abaixo dela tem suas tabelas e *views*. E na direita da

imagem, é onde se realizam as *queries*, tanto para aplicações analíticas quanto aplicações de *machine learning* (Lakshmanan, 2019).

Figura 23 – Console do BigQuery



Fonte: Autoria própria.

3.3.3 BigQuery ML

BigQuery ML possibilita criar e rodar modelos de aprendizado de máquina utilizando consultas em GoogleSQL. Os modelos de BigQuery ML são armazenados nos conjuntos de dados, semelhante às tabelas e *views*. Há muitas vantagens em utilizá-lo, como:

democratização do uso de Machine Learning e Inteligência Artificial empoderando analista de dados, usuários primários de banco de dados, construindo e criando modelos utilizando as existentes ferramentas de inteligência de negócios; não é necessário utilizar linguagem como Java e Python, é possível treinar modelos e acessar recursos de IA utilizando SQL, a linguagem mais familiar entre os analistas de dados; a velocidade de desenvolvimento e inovação por não precisar ficar movimentando os dados (Lakshmanan, 2019).

Para se criar um modelo, basta executar a *query* de criação de modelo com seus respectivos parâmetros (Google, 2025).

Quadro 1 – Código da criação do modelo no BigQuery

```
      {CREATE MODEL | CREATE MODEL IF NOT EXISTS | CREATE OR REPLACE MODEL}
model_name
      OPTIONS(model_option_list)
      AS query_statement

      model_option_list:
      MODEL_TYPE = { 'DNN_CLASSIFIER' | 'DNN_REGRESSOR' }
               [, LEARN_RATE = { float64_value | HPARAM_RANGE(range) |
HPARAM_CANDIDATES([candidates]) } ]
          [, OPTIMIZER = { { 'ADAGRAD' | 'ADAM' | 'FTRL' | 'RMSPROP' | 'SGD'
} | HPARAM_CANDIDATES([candidates]) } ]
                [, L1_REG = { float64_value | HPARAM_RANGE(range) |
HPARAM_CANDIDATES([candidates]) } ]
                [, L2_REG = { float64_value | HPARAM_RANGE(range) |
HPARAM_CANDIDATES([candidates]) } ]
          [, ACTIVATION_FN = { { 'RELU' | 'RELU6' | 'CRELU' | 'ELU' | 'SELU'
| 'SIGMOID' | 'TANH' } | HPARAM_CANDIDATES([candidates]) } ]
                [, BATCH_SIZE = { int64_value | HPARAM_RANGE(range) |
HPARAM_CANDIDATES([candidates]) } ]
                [, DROPOUT = { float64_value | HPARAM_RANGE(range) |
HPARAM_CANDIDATES([candidates]) } ]
                [, HIDDEN_UNITS = { int_array | HPARAM_RANGE(range) |
HPARAM_CANDIDATES([candidates]) } ]
          [, INTEGRATED_GRADIENTS_NUM_STEPS = int64_value ]
          [, TF_VERSION = { '1.15' | '2.8.0' } ]
          [, AUTO_CLASS_WEIGHTS = { TRUE | FALSE } ]
          [, CLASS_WEIGHTS = struct_array ]
          [, ENABLE_GLOBAL_EXPLAIN = { TRUE | FALSE } ]
          [, EARLY_STOP = { TRUE | FALSE } ]
```

```
          [, MIN_REL_PROGRESS = float64_value ]
          [, INPUT_LABEL_COLS = string_array ]
          [, MAX_ITERATIONS = int64_value ]
          [, WARM_START = { TRUE | FALSE } ]
            [, DATA_SPLIT_METHOD = { 'AUTO_SPLIT' | 'RANDOM' | 'CUSTOM' |
'SEQ' | 'NO_SPLIT' } ]
          [, DATA_SPLIT_EVAL_FRACTION = float64_value ]
          [, DATA_SPLIT_TEST_FRACTION = float64_value ]
          [, DATA_SPLIT_COL = string_value ]
          [, NUM_TRIALS = int64_value ]
          [, MAX_PARALLEL_TRIALS = int64_value ]
           [, HPARAM_TUNING_ALGORITHM = { 'VIZIER_DEFAULT' | 'RANDOM_SEARCH'
| 'GRID_SEARCH' } ]
          [, HPARAM_TUNING_OBJECTIVES = { 'ROC_AUC' | 'R2_SCORE' | ... } ]
          [, MODEL_REGISTRY = { 'VERTEX_AI' } ]
          [, VERTEX_AI_MODEL_ID = string_value ]
          [, VERTEX_AI_MODEL_VERSION_ALIASES = string_array ]
          [, KMS_KEY_NAME = string_value ]
```

Fonte: Google Cloud (2025).

onde:

- CREATE MODEL: Cria e treina o modelo em um *dataset* específico. Se o modelo já existe, irá retornar um erro.
- CREATE MODEL IF NOT EXISTS: Cria e treina o modelo apenas se o modelo não existir no *dataset* específico.
- CREATE OR REPLACE MODEL: Cria e treina o modelo e substitui o modelo existente com o mesmo nome no *dataset* especificado.
- model_name: É o nome do modelo que está sendo criado ou substituído.
- MODEL_TYPE: Especifica o tipo do modelo, se será de regressão ou classificação.
- LEARN_RATE: A taxa de aprendizado inicial para o treino.
- OPTIMIZER: O algoritmo de otimização para treinar o modelo. Existe o “ADAM” (Algoritmo ADAM), “ADAGRAD” (Algoritmo Adagrad), “FTRL” (Algoritmo FTRL), “RMSPROP” (Algoritmo RMSProp) e “SGD” (Algoritmo descida do gradiente). Valor padrão é “ADAM”.
- L1_REG: A força da regularização L1 no otimizador. Valor padrão é 0.
- L2_REG: A força da regularização L2 no otimizador. Valor padrão é 0.
- ACTIVATION_FN: A função de ativação na rede neural. Há os tipos “RELU” (Linea retificada), “RELU6” (Linear retificada 6), “CRELU” (ReLU concatenada), “ELU”

(Linear exponencial), “SELU” (Linear exponencial escalar), “SIGMOID” (Sigmoid), “TANH” (Tanh). Valor padrão é “RELU”.

- BATCH_SIZE: O tamanho do mini-lote das amostras que são alimentadas à rede neural. Valor padrão é 32.
- DROPOUT: A taxa da desativação aleatória das unidades na rede neural. Valor padrão é 0.
- HIDDEN_UNITS: São as camadas ocultas da rede neural. Caso não seja especificado, o BigQuery ML aplica uma única camada oculta que contém não mais que 128 unidades. O número é calculado como [min(128, num_samples / (10 * (num_input_units + num_output_units)))].
- INTEGRATED_GRADIENTS_NUM_STEPS: Específica o número de passos para amostragem entre o exemplo sendo explicado e sua referência para aproximar a integral quando utilizar métodos de atribuição de gradientes integrais. O valor padrão é 15.
- TF_VERSION: Especifica a versão do TensorFlow para o treinamento do modelo. O valor padrão é 1.15.
- AUTO_CLASS_WEIGHTS: Determina se é necessário balancear o número de classes utilizando os pesos de cada classe sendo inversamente proporcional à frequência daquela classe. O valor padrão é “FALSE”.
- CLASS_WEIGHTS: Peso de cada classe.
- ENABLE_GLOBAL_EXPLAIN: Determina se irá utilizar a computação global utilizando *explainable AI* para avaliar a importância das *features* do modelo. O valor padrão é FALSE.
- EARLY_STOP: Determina se é necessário parar após a primeira iteração em que a perda é menor que o valor especificado em MIN_REL_PROGRESS. O valor padrão é “TRUE”.
- MIN_REL_PROGRESS: A perda mínima relativa necessária para continuar o treinamento do modelo. O valor padrão é 0.01.
- INPUT_LABEL_COLS: O nome da coluna de *output* do modelo. Por exemplo, a indicação se é spam ou não. O padrão irá puxar a coluna com o nome de *label*.
- MAX_INTERATIONS: O número máximo de iterações para o treinamento do modelo. O valor padrão é 20.
- WARM_START: Determina se é necessário treinar o modelo com novos dados de treino, com novas opções, ou ambos. O valor padrão é “FALSE”.

- DATA_SPLIT_METHOD: O método utilizado para separar os dados em dados de treinamento, avaliação. “AUTO_SPLIT” é o padrão em que caso tenha menos que 500 linhas, todas as linhas serão utilizadas para treinamento. Se não estiver utilizando ajuste de hiperparâmetros, se houver entre 500 e 50000 linhas, 20% será utilizado para avaliação dos dados e 80% para treinamento e se houver mais de 50000 linhas, 10000 serão utilizados para avaliação e as linhas remanescentes serão utilizados para treinamento. Se estiver utilizando ajuste de hiperparâmetros, 10% será utilizado para avaliação, 10% será utilizado para teste e 80% para treinamento.
- DATA_SPLIT_EVAL_FRACTION: A fração dos dados para avaliação do modelo. O valor padrão é 0.2.
- DATA_SPLIT_TEST_FRACTION: A fração dos dados para teste do modelo. O valor padrão é 0.
- DATA_SPLIT_COL: Nome da coluna usada para ordenar os dados de entrada para o treinamento, avaliação ou teste.
- NUM_TRIALS: O número máximo de submodelos para treinamento.
- MAX_PARALLEL_TRIALS: O número máximo de treinamentos para rodar ao mesmo tempo.
- HPARAM_TUNING_ALGORITHM: O algoritmo utilizado para ajustar os hiperparâmetros. O valor padrão é o “VIZIER_DEFAULT”, o qual é um algoritmo do Vertex AI Vizier para ajustar os parâmetros em que se mistura algoritmos de busca, incluindo otimização bayesiana com processo gaussiano.
- HPARAM_TUNING_OBJECTIVES: O objetivo de ajuste de hiperparâmetro para o modelo.

Os parâmetros seguintes são de infraestrutura, caso queira colocar o modelo no Vertex AI, um serviço da Google de IA. E assim, sabendo utilizar essa função corretamente, será criado o modelo de rede neural para os dados treinados.

Após visto como que se cria um modelo, é necessário saber como que o avalia e como se realiza predições a partir dela. O Quadro 2 abaixo, mostra como avaliar o modelo (Google, 2025):

Quadro 2 – Código da avaliação do modelo no BigQuery

```
# All other types of models:
ML.EVALUATE(
  MODEL `project_id.dataset.model`
  [, { TABLE `project_id.dataset.table` | (query_statement) }],
    STRUCT(
      [threshold_value AS threshold]
      [, trial_id AS trial_id])
)
```

Fonte: Google Cloud (2025).

onde:

- project_id: O ID do projeto.
- dataset: O *dataset* do BigQuery que contém o modelo.
- model: Nome do modelo.
- table: Nome da tabela de *inputs* que o modelo. Caso não seja passada, irá mostrar as métricas de avaliação dos dados de treino.
- query_statement: *Query* que seja usada para gerar os dados de avaliação. Caso não seja passada, irá mostrar as métricas de avaliação dos dados de treino.
- threshold: Um valor real que especifica o limite para classificações binárias. O valor padrão é 0,5.
- trial_id: Um valor inteiro que define o teste de ajuste de hiperparâmetro que a função deve-se avaliar.

E depois de avaliar o modelo, deve-se realizar as predições com a função do Quadro 3 abaixo (Google, 2025).

Quadro 3 – Código da predição do modelo no BigQuery

```
ML.PREDICT(
  MODEL `project_id.dataset.model`,
  { TABLE `project_id.dataset.table` | (query_statement) },
  STRUCT(
    [threshold_value AS threshold]
    [, keep_columns AS keep_original_columns]
    [, trial_id AS trial_id])
)
```

Fonte: Google Cloud (2025).

3.4 Percentis

3.4.1 Conceito de percentil

A maneira mais comum de se relatar a classificação relativa de um número dentro de um *dataset* é utilizando percentis. Um percentil é a porcentagem de indivíduos de um *dataset* que estão abaixo de onde seu número particular é localizado. Por exemplo, se a nota de um exame e a posição é de 90º percentil, significa que 90% das pessoas que fizeram o exame ficaram com as notas menores e 10% ficaram com notas maiores.

Para se calcular o percentil kº (onde k é o número entre um e cem), deve-se fazer os seguintes passos:

1. Ordenar todos os números do menor para o maior, ou o contrário, depende da aplicação que se deseja.
2. Multiplicar k% vezes o número total de linhas/números n.
3. Se no passo 2 resultar em um número inteiro, vá para o passo 5. Se o resultado do passo 2 não for um número inteiro, arredonde para o número inteiro seguinte mais próximo e vá para o passo 4.
4. Contar os números do *dataset* do começo ao fim até alcançar o valor do passo 3. O número correspondente é o percentil kº.
5. Contar os números do *dataset* do começo ao fim até alcançar o número inteiro. O kº percentil é a média do número correspondente do *dataset* e do próximo número do *dataset.*

Por exemplo, há 25 notas de um exame, do menor ao maior: 43, 54, 56, 61, 62, 66, 68, 69, 69, 70, 71, 72, 77, 78, 79, 85, 87, 88, 89, 93, 95, 96, 98, 99, 99. Para achar o percentil 90º dessas notas, deve-se começar multiplicando 90% pelo número de notas, o que dá 90%*25 = 22,5 (Passo 2). Isso não é um número inteiro, logo deve-se arredondar para cima, o qual é 23. O 23º número dessa lista é 98, então esse número é o percentil 90º deste *dataset*.

Outro exemplo, caso se deseje o percentil 20º, então 20%*25 = 5. Isso é um número inteiro, desse modo deve-se ir ao passo 4, em que deve-se fazer a média dos número da posição e da próxima, resultando em (62+66)/2 = 64. Seguindo essa lógica, a mediana é o 13º número, o qual é o 77 (Rumsey, 2010).

3.4.2 O resumo dos cinco números

O resumo dos cinco números é um conjunto de cinco valores-chave que dividem o *dataset* em quatro seções iguais. Os cinco números do resumo são:

1. O menor número do *dataset*.
2. O percentil 25º, ou primeiro quartil, ou Q1.
3. A mediana (ou percentil 50º).
4. O percentil 75º, ou terceiro quartil, ou Q3.
5. O maior número do *dataset*.

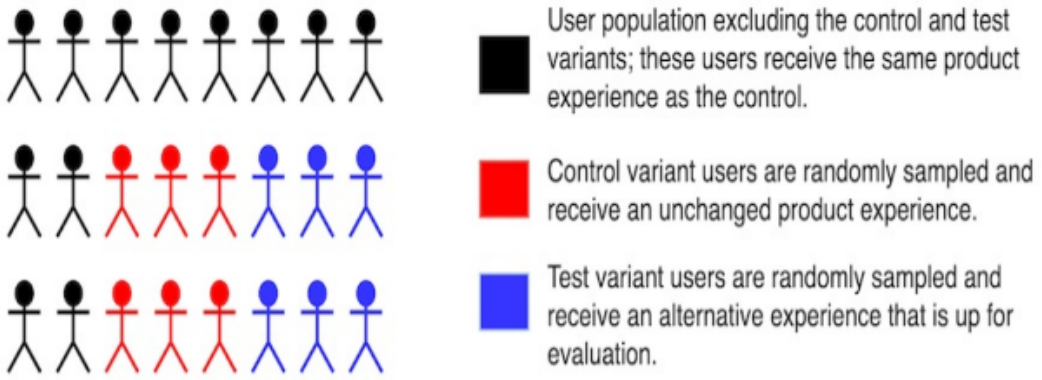
No mesmo exemplo anterior, para se achar o percentil 25º, deve-se realizar a operação 25%*25 = 6,25, o número inteiro acima é o número 7, e o 7º número do *dataset* é 68. Para se achar Q3, deve-se realizar a operação 75%*25 = 18,75, o número inteiro acima é o número 19, e o 19º número do *dataset* é 89. Colocando todos os valores juntos se obtém o resumo de 43, 68, 77, 89 e 99.

Com isso se dá para separar em quatro grupos, Mínimo-Q1, Q1-Q2, Q2-Q3 e Q3-Máximo. Em um outro exemplo, caso tenha uma ordenação de probabilidade de uma pessoa comprar, será muito benéfico realizar essa classificação em 4 grupos, pois irá facilitar no tipo de relacionamento que irá se desejar com o grupo (Rumsey, 2010).

3.5 Teste A/B

Teste A/B é um experimento de controle que mede o impacto de uma mudança em um subconjunto de usuários. Um teste A/B efetivo é onde há confiança nas decisões baseadas nos resultados. Esse teste, em sua forma mais simples, consiste em dois grupos de usuários: o primeiro grupo é a variante de controle, que inclui os usuários que recebem a funcionalidade ou experiência do produto sem alterações; o segundo grupo é composto por usuários que recebem um novo recurso ou mudança que está sendo avaliada, conhecido como a variante de teste. Tanto a variante de teste quanto a de controle precisam ser aleatórias para garantir que os usuários atribuídos a cada grupo sejam estatisticamente semelhantes, permitindo maior confiança na medição do efeito do experimento.

Figura 24 – Separação dos grupos em variante de controle e variante de teste



Fonte: Nassery (2023).

Depois de entendido como se funciona o teste A/B, deve-se saber se o teste foi um sucesso. A seguinte lista de checagem ajuda a como começar a validar:

1. Identifique a mudança para a avaliação. Ela pode ser uma mudança de UX, uma funcionalidade nova ou uma migração de sistemas de engenharia.
2. Crie um estado de hipótese. Com ela irá responder no que se acredita que a mudança irá fazer de diferente, no que ela resultará entregando a definição de sucesso e por fim a razão dessa mudança.
3. Selecione a métrica de sucesso. Se a métrica de sucesso inicialmente selecionada não for possível ser computada, considere métricas substitutas.
4. Defina as métricas de segurança para o experimento. Métricas que precisam ser monitoradas, mas não otimizadas.
5. Crie as variantes de teste e controle. Certifique-se de que a composição das variantes seja amostrada aleatoriamente da população geral (Nassery, 2023).

3.6 Visualização de dados

O sucesso de uma visualização dos dados não se começa pela visualização dos dados. O mais importante é entender o contexto para quem quer se comunicar. Quando se trata de análise explicativa, ser capaz de articular de forma concisa para quem se deseja comunicar e o que deseja transmitir antes de começar a construir o conteúdo, reduz o número de iterações e ajuda a garantir que a comunicação atenda ao propósito pretendido.

Há tipos de visualizações que são mais utilizadas. Em vários casos, não existe a visualização correta, ao contrário, frequentemente existem diferentes tipos de gráficos que podem atender a necessidade. Se houver dúvidas de qual é o melhor gráfico nessa situação,

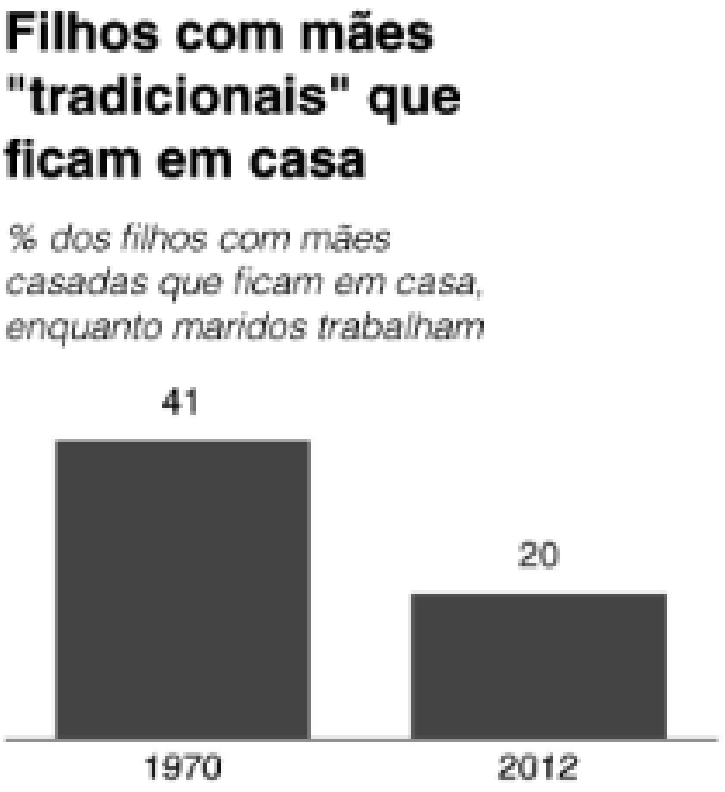
basta lembrar que deve ser o que a audiência achar mais fácil de entender. A seguir serão apresentados tipos de gráficos e suas principais características e quando utilizá-las (Knaflic, 2019).

3.6.1 Tipos de Gráficos

- Texto simples

Quando há um número ou dois para compartilhar, texto simples pode ser uma ótima maneira de se comunicar. Com poucas palavras de suporte para clarificar os pontos, poderá evitar interpretações erradas.

Figura 25 – Gráfico original das mães que ficam em casa



Fonte: Knaflic (2019).

O fato de ter números, não necessariamente deve-se fazer um gráfico, a figura abaixo retrata como apenas o texto ajuda na interpretação dos dados (Knaflic, 2019).

Figura 26 – Remodelação do gráfico original das mães que ficam em casa

Fonte: Knaflic (2019).

- Tabelas

As tabelas interagem com o sistema verbal humano, ou seja, os humanos leem os dados. Quando se há tabelas, é natural que se leiam as linhas e colunas e comparem os valores. O design deve assumir um papel secundário e os dados em primeiro plano (Knaflic, 2019).

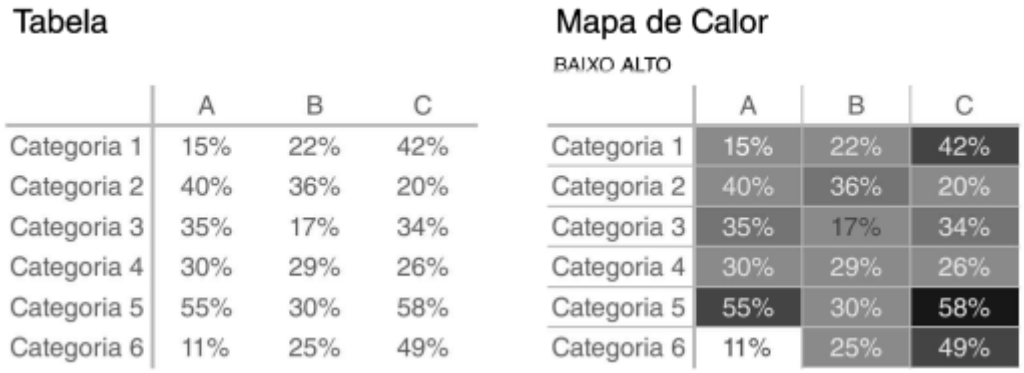
Figura 27 – Bordas de tabela

Fonte: Knaflic (2019).

- Mapa de calor

Mapas de calor é um modo de visualizar dados em formato tabular, em vez dos números, são utilizadas células coloridas que transmitem a magnitude relativa do número.

Figura 28 – Diferenciação entre tabela e mapa de calor



	A	B	C
Categoria 1	15%	22%	42%
Categoria 2	40%	36%	20%
Categoria 3	35%	17%	34%
Categoria 4	30%	29%	26%
Categoria 5	55%	30%	58%
Categoria 6	11%	25%	49%

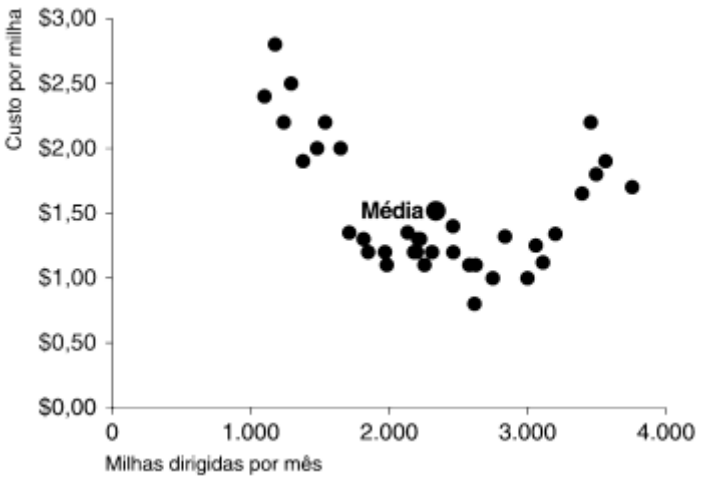
Fonte: Knaflic (2019).

Na imagem anterior, nota-se que na tabela da esquerda é muito mais difícil entender quais números são altos e quais são baixos. Para reduzir esse processo mental, pode-se utilizar saturação de cor, o que ajuda o olho humano a atingir mais rapidamente os pontos de interesse (Knaflic, 2019).

- Dispersão

Os gráficos de dispersão podem ser úteis para mostrar a relação entre duas variáveis, pois permitem codificar dados simultaneamente em um eixo x horizontal e em um eixo y vertical para mostrar a relação existente entre os dois. Tende-se a ser utilizado nos campos científicos.

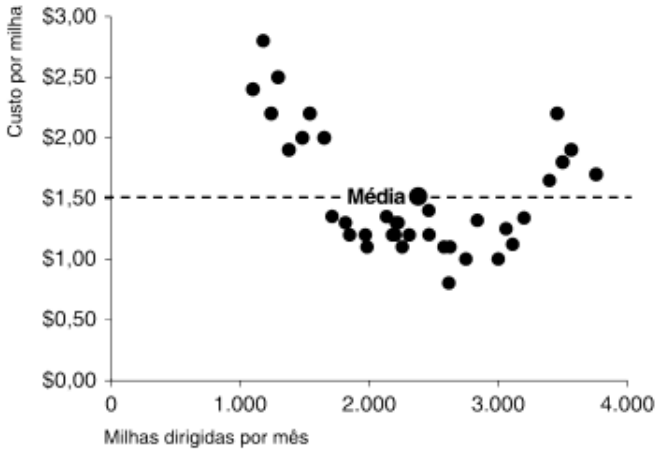
Figura 29 – Custo por milha por milhas dirigidas



Fonte: Knaflic (2019).

Nesse gráfico, se utilizar a média, os casos nos quais o custo por milha está acima da média fica muito mais simples de visualizar (Knaflic, 2019).

Figura 30 – Custo por milha por milhas dirigidas com a linha da média

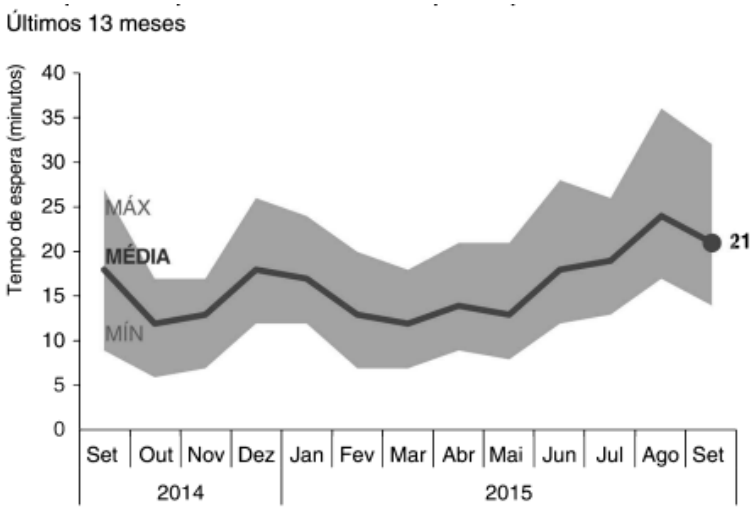


Fonte: Knaflic (2019).

- Linhas

Os gráficos de linhas são mais utilizados para registrar dados contínuos. Os pontos são fisicamente conectados por meio da linha, indicando uma conexão, sendo comumente utilizado em dados que estão em alguma unidade de tempo: dias, meses, trimestres ou anos. Em alguns casos a linha pode representar um resumo estatístico, como a média (Knaflic, 2019).

Figura 31 – Tempo de espera no controle de passaportes

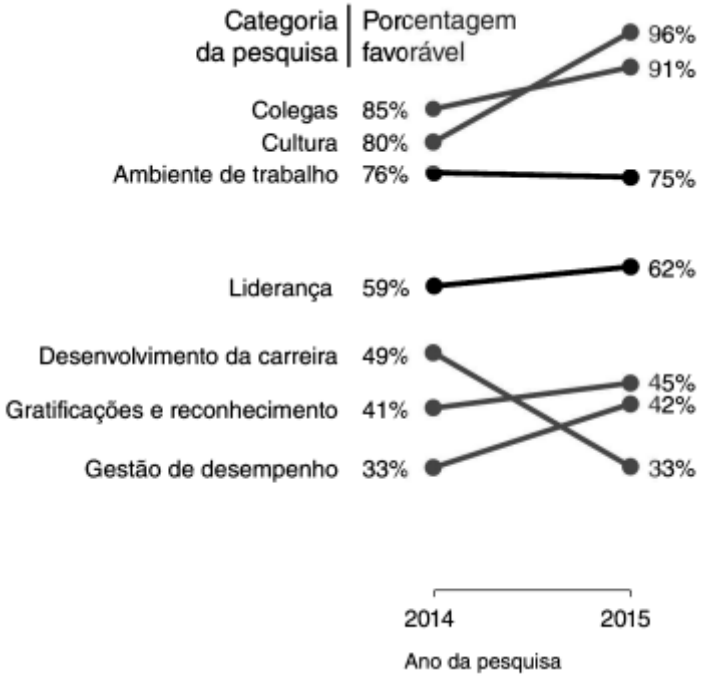


Fonte: Knaflic (2019).

- Inclinação

Esses gráficos são úteis quando se tem dois períodos de tempo ou pontos de comparação e deseja mostrar rapidamente os aumentos ou diminuições relativas ou diferenças de várias categorias entre os dois pontos de dados. Esses gráficos contêm muitas informações, além dos valores absolutos, as linhas que conectam fornecem o aumento ou decréscimo visual na taxa de mudança (Knaflic, 2019).

Figura 32 – Opinião dos funcionários ao longo do tempo

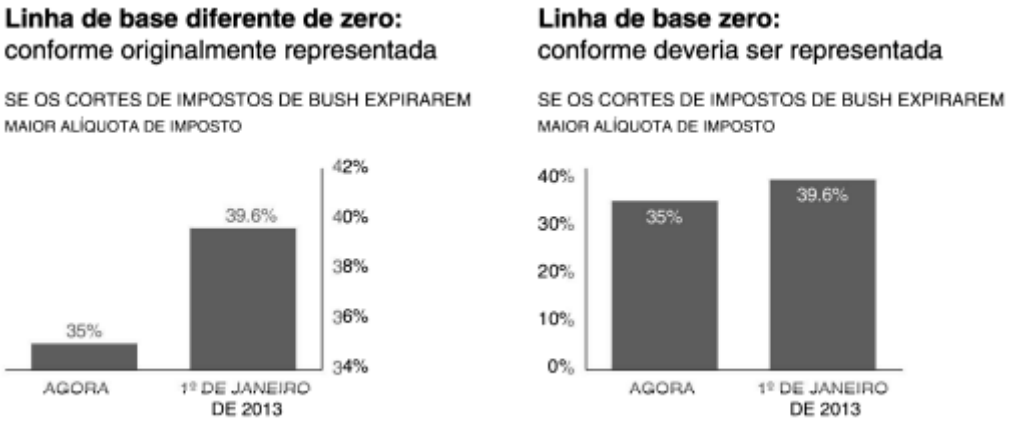


Fonte: Knaflic (2019).

- Barras verticais

Os gráficos de barras são evitados por serem comuns, mas isso é um erro. O público precisa de menos curva de aprendizado e utiliza o gráfico para decidir qual informação deseja extrair da apresentação. Os olhos leem gráficos de barras facilmente, realizam comparações nos pontos extremos das barras, em que rapidamente conseguem visualizar qual categoria é maior, qual é menor e a diferença entre as categorias (Knaflic, 2019).

Figura 33 – Gráficos de barra com e sem linha de base zero

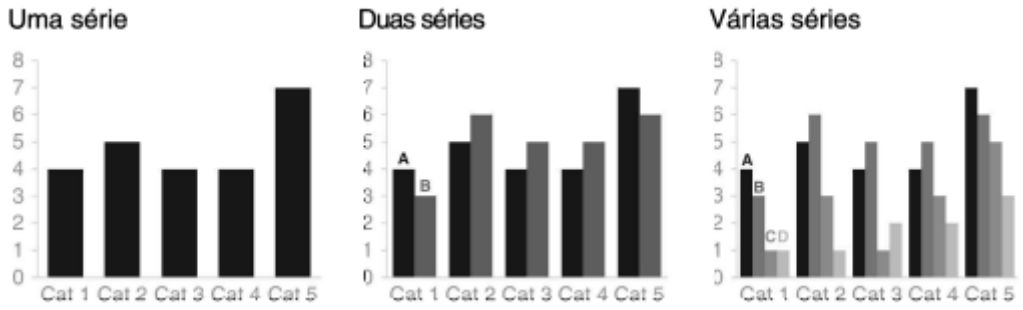


Fonte: Knaflic (2019).

Na figura acima, um enorme aumento à esquerda é consideravelmente reduzido quando representado adequadamente. Nesse gráfico, pode-se notar que o aumento de imposto não é tão preocupante quanto originalmente representado, por isso é aconselhável sempre utilizar a linha na base zero, para evitar essas distorções. Essa regra não se aplica aos gráficos de linhas, pois o enfoque do gráfico de linha é mostrar as posições relativas no espaço.

Os gráficos de barras verticais ou gráficos de colunas podem ter mais de uma série, e à medida que séries de dados são adicionadas, mais difícil é de entender. É importante considerar que ocorre um agrupamento visual, logo deve-se ter uma ordem relativa da categorização para melhor entendimento (Knaflic, 2019).

Figura 34 – Gráficos de barras verticais



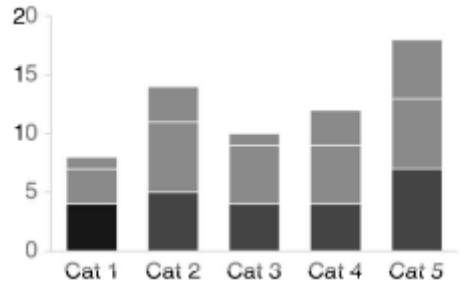
Fonte: Knaflic (2019).

- Barras verticais empilhadas

Os casos de usos para gráficos de barras verticais empilhadas são para comparar o total entre categorias e visualizar as partes dos subcomponentes dentro de determinada

categoria. Esse gráfico pode ser estruturado como números absolutos ou com cada coluna somando 100% (Knaflic, 2019).

Figura 35 – Gráficos de barras empilhadas

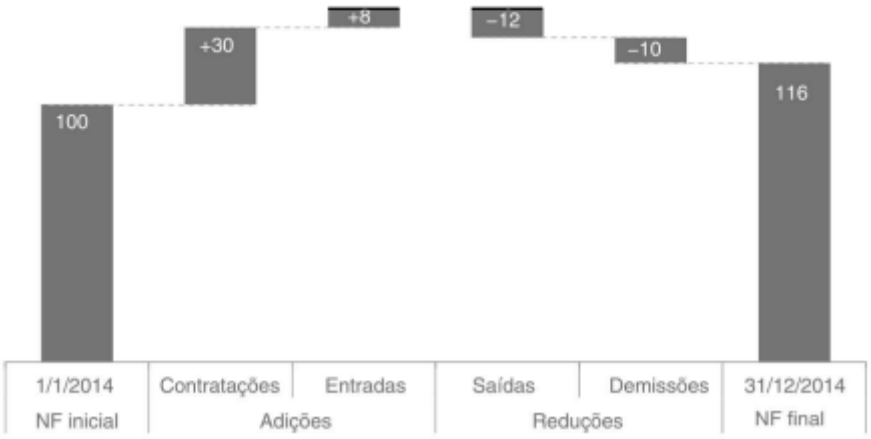


Fonte: Knaflic (2019).

- Cascata

O gráfico de cascata pode ser usado para separar as partes de um gráfico de barras empilhadas com finalidade de focar uma por vez ou para mostrar um ponto de partida, aumentos ou reduções e o ponto final resultante.

Figura 36 – Número de funcionários em gráficos de cascata



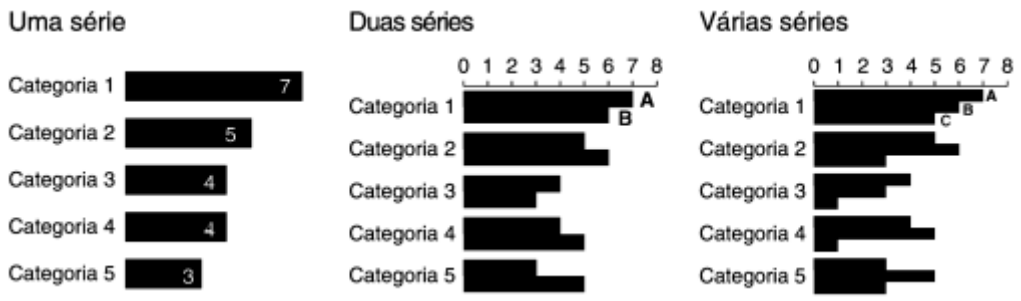
Fonte: Knaflic (2019).

Nesse gráfico, no lado esquerdo, o número de funcionários na equipe no início do ano, quando ir para o lado direito, há as ações incrementais ou redutivas e a última coluna representa o número total de funcionários no final do ano (Knaflic, 2019).

- Barra horizontal

Esse é o melhor gráfico para dados categóricos, pois ele é extremamente fácil de ler. Ele pode ser útil quando o nome das categorias são longas, pois os textos são escritos da esquerda para a direita. Além disso, por causa desse modo de leitura, a estrutura de gráfico de barras horizontais é o modo em que os olhos enxergam primeiramente as categorias e logo em seguida os dados. E assim como o gráfico de barra vertical, o horizontal pode ter mais de uma série também (Knaflic, 2019).

Figura 37 – Gráfico de barras com uma ou mais séries

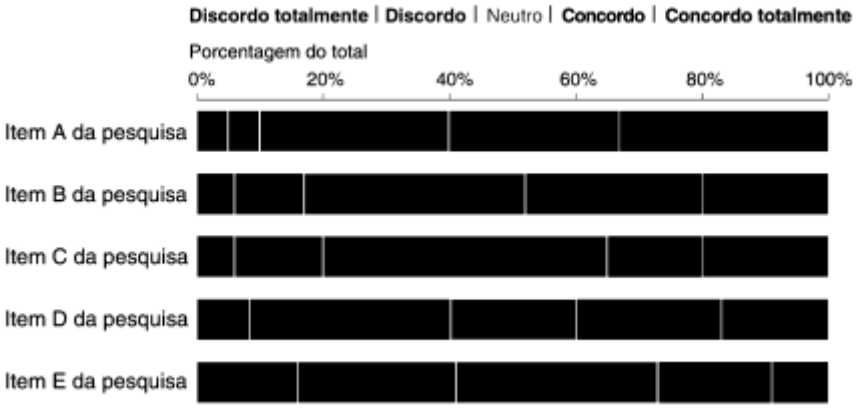


Fonte: Knaflic (2019).

- Barras horizontais empilhadas

Semelhantes ao gráfico de barras verticais empilhadas, os gráficos de barras horizontais também servem para mostrar os totais em diferentes categorias, dar ideia de subcomponentes e podem ser estruturados de forma a mostrar os valores absolutos ou até a soma 100% (Knaflic, 2019).

Figura 38 – Gráfico de barras horizontais empilhadas a 100%

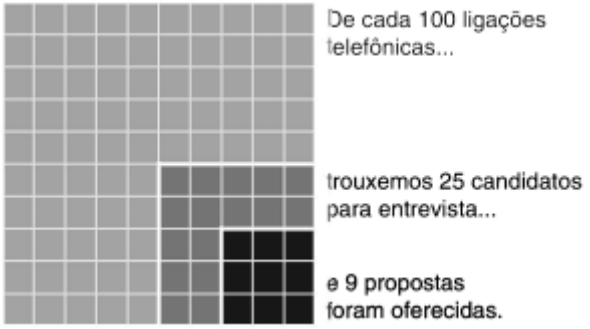


Fonte: Knaflic (2019).

- Área

Esse gráfico é bom de ser evitado, pois os olhos humanos não conseguem atribuir bem o valor quantitativo no espaço bidimensional, dificultando a leitura desses tipos de gráfico. Esse gráfico é bom para ser utilizado quando se usa um quadrado que tem altura e largura (Knaflic, 2019).

Figura 39 – Gráfico de área

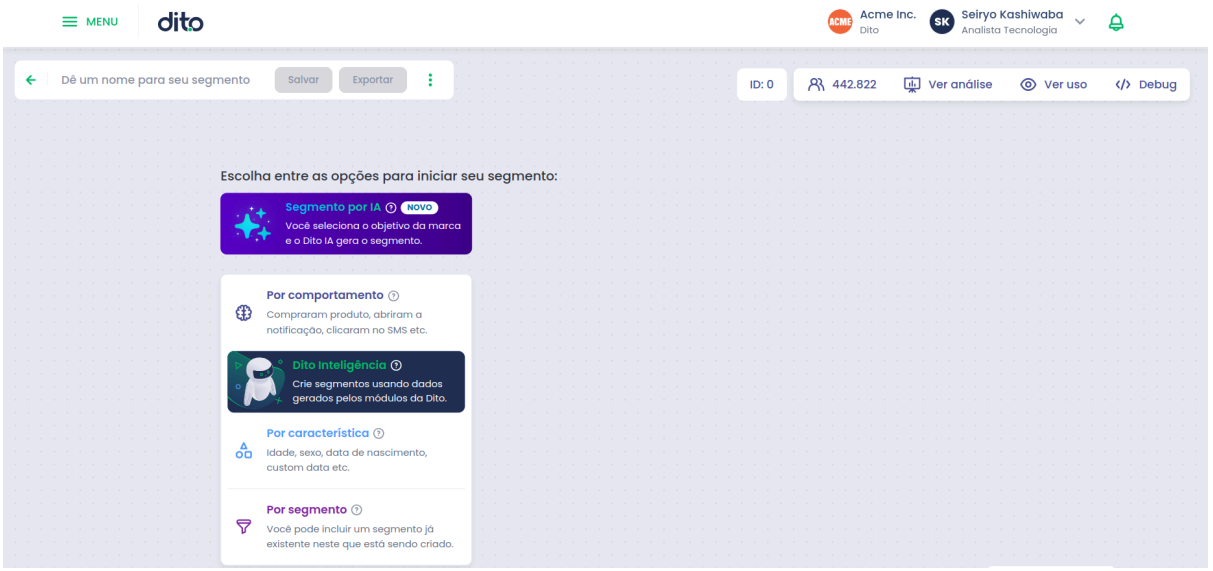


Fonte: Knaflic (2019).

3.7 Dito CRM

Dito CRM é uma empresa *Software as a Service* (SaaS), ou seja, seu produto é um software, em que seu foco é criar estratégias centradas nas pessoas para oferecer produtos inovadores. Dito Campanhas é o produto onde se pode realizar disparos personalizados por meio dos atributos e características dos consumidores.

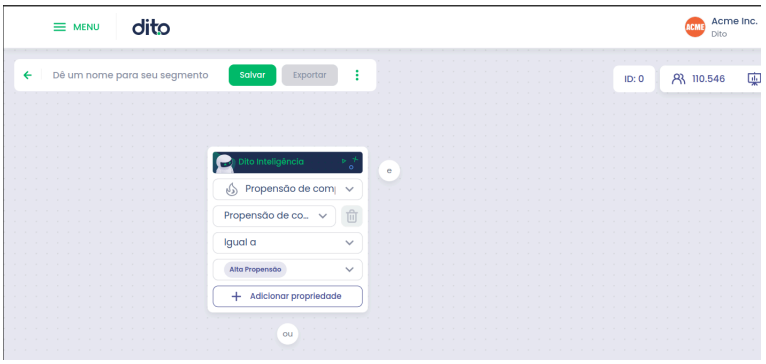
Figura 40 – Tela de segmentação dos consumidores no aplicativo da Dito



Fonte: Autoria própria.

Após o treinamento do modelo, as classificações dos consumidores ficam armazenadas em “Dito Inteligência” e com a característica “Propensão de compra” preenchida. Depois deve-se dar um nome ao segmento e em seguida salvá-lo.

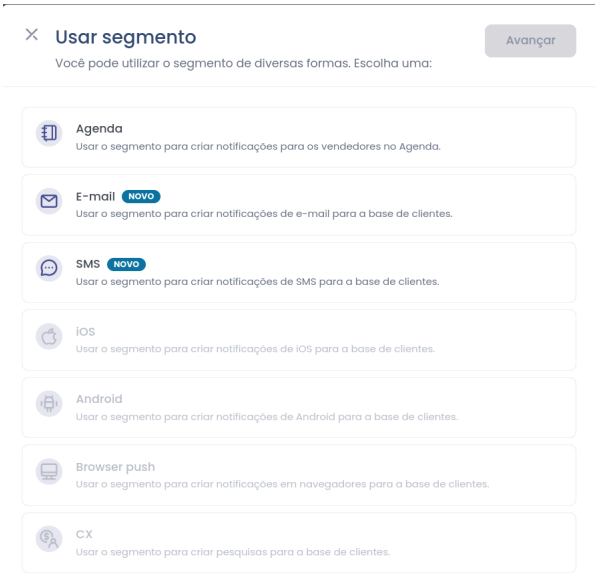
Figura 41 – Segmentação com “Propensão de compra”



Fonte: Autoria própria.

Depois de criado o segmento, basta utilizá-lo, clicando nos três pontos ao lado de “Exportar” e selecionar “Usar segmento”. A Figura abaixo mostra as opções de disparo do segmento criado, as quais são: Agenda (produto para o vendedor contatar via WhatsApp), E-mail, SMS, iOS (notificação em sistemas iOS), Android (notificação em sistemas Android), Browser push (notificação em navegadores) e CX (produto de envio de pesquisas de satisfação).

Figura 42 – Opção de disparo de campanha

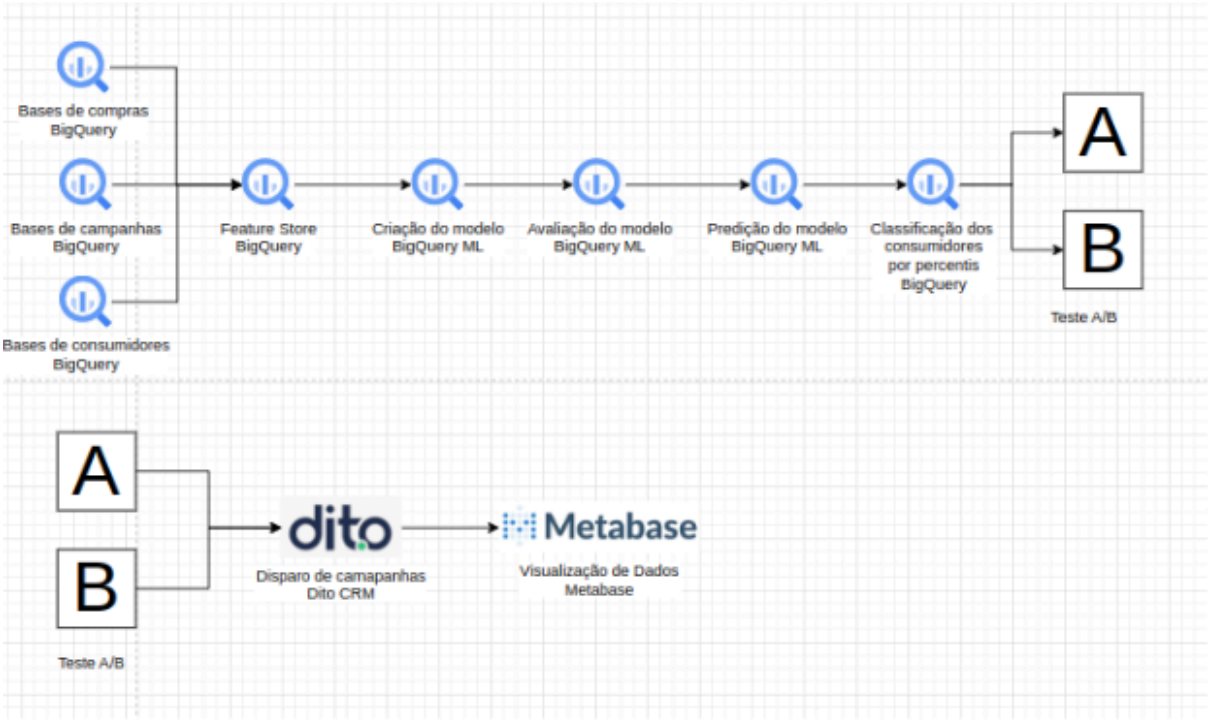


Fonte: Autoria própria.

4 METODOLOGIA

Na seção anterior, foi mostrado toda a teoria para poder se realizar o projeto. Nesta seção será mostrado como foi feita a junção de todos os conceitos para a aplicação de uso de *Machine Learning* na área de CRM. Deve-se ter contexto do que é CRM e com isso realizar um estudo exploratório dos dados existentes para poder verificar quais serão as variáveis importantes para o modelo. Tendo em vista que o foco desse projeto é realizar mais vendas, então o *output* desse modelo deverá ser a indicação se o cliente irá comprar ou não, de acordo com a sua pontuação de compra. As subseções mostrarão o passo a passo de como todo o projeto foi feito. O diagrama da figura a seguir apresenta todo o procedimento do projeto.

Figura 43 – Diagrama do procedimento experimental do projeto



Fonte: Autoria própria.

4.1 Criação da feature store

A *feature store* foi feita no BigQuery, com a granularidade de semana do status do consumidor, marca e código do identificador do consumidor, com seus atributos de vendas e interações. Foram utilizados os seguintes atributos:

- agenda_influenced_purchases_before_contact: Quantidade de compras influenciadas pelo produto Agenda.

- agenda_sum_influenced_revenue_before_contact: Soma da receita influenciada pelo produto Agenda.
- agenda_avg_window_hours_before_contact: Média de horas para se realizar a compra influenciada pelo produto Agenda.
- agenda_contact_window_weeks: Semanas sem receber contato com o produto Agenda.
- agenda_purchases_window_weeks: Semanas sem compra influenciada pelo produto Agenda.
- sms_influenced_purchases_before_contact: Quantidade de compras influenciadas pelo disparo de SMS.
- email_influenced_purchases_before_contact: Quantidade de compras influenciadas pelo disparo de Email.
- sms_sum_influenced_revenue_before_contact: Soma da receita influenciada pelo disparo de SMS.
- email_sum_influenced_revenue_before_contact: Soma da receita influenciada pelo disparo de Email.
- sms_avg_window_hours_before_contact: Média de horas para se realizar a compra influenciada pelo disparo de SMS.
- email_avg_window_hours_before_contact: Média de horas para se realizar a compra influenciada pelo disparo de Email.
- sms_dispatch_window_weeks: Semanas sem receber disparos de SMS .
- email_dispatch_window_weeks: Semanas sem receber disparos de Email.
- email_open_window_weeks: Semanas sem abertura de emails.
- sms_purchases_window_weeks: Semanas sem compra influenciada pelo disparo de SMS.
- email_purchases_window_weeks: Semanas sem compra influenciada pelo disparo de Email.
- cnt_open_notification_last_week: Quantidade de aberturas de email na semana anterior da semana de referência.
- cnt_abandoned_cart_last_week: Quantidade de carrinho abandonado na semana anterior da semana de referência.
- cnt_access_product_last_week: Quantidade de acessos ao produto na semana anterior da semana de referência.

- cnt_access_cart_last_week: Quantidade de acessos ao carrinho na semana anterior da semana de referência.
- cnt_access_home_last_week: Quantidade de acessos à pagina incial na semana anterior da semana de referência.
- cnt_open_notification_last_month: Quantidade de aberturas de email no mês anterior da semana de referência.
- cnt_abandoned_cart_last_month: Quantidade de carrinho abandonado no mês anterior da semana de referência.
- cnt_access_product_last_month: Quantidade de acessos ao produto no mês anterior da semana de referência.
- cnt_access_cart_last_month: Quantidade de acessos ao carrinho no mês anterior da semana de referência.
- cnt_access_home_last_month: Quantidade de acessos à pagina incial no mês anterior da semana de referência.
- accumulated_revenue: Soma da receita acumulada do consumidor.
- accumulated_purchases: Soma do número de compras acumuladas do consumidor.
- accumulated_purchases_last_month: Soma do número de compras acumuladas até o mês anterior.
- accumulated_purchases_last_year: Soma do número de compras acumuladas até o ano anterior
- purchases_window_weeks: Semanas sem compras do consumidor.
- ct_success_purchases: Número de compras na semana de referência de acordo com os atributos anteriores.

Com essas variáveis, é possível realizar o treinamento do modelo, com seus respectivos parâmetros. Essa é a etapa mais complicada de todo o processo, pois deve-se entender como os dados estão organizados, quais modificações são necessárias a serem realizadas, o que se espera do resultado, qual modelo deve-se utilizar. Então é normal que essa etapa seja a que mais levará tempo a ser desenvolvida.

4.2 Treinamento, avaliação e predição do modelo

Seguindo os passos da subseção 3.2.6:

1. A métrica definida é o *recall* pois foca nos falsos negativos, ou seja, a parte em que houve compras e não foram computadas como positivos estão no denominador,

diminuindo o valor da porcentagem. No contexto de otimização de vendas, as campanhas enviadas devem ter o maior número de conversão, ou seja, é necessário maximizar essa métrica pois a ideia não é perder um cliente que estava propenso a comprar e ele não ser identificado. E para evitar confusão, pois sempre há confusão entre *recall* e precisão, a precisão teria um foco maior nas predições, ou seja, entre todos os clientes que o modelo classificou como propenso a comprar, quantos realmente compraram.

2. Não há uma evidência explícita que o BigQuery utilize a métrica *binary categorical cross-entropy*, porém como o modelo é baseado em TensorFlow. A função de custo padrão é a *binary categorical cross-entropy*, visto nas equações 3 e 4.
3. Também não há uma explicitude para a estratégia de inicialização dos parâmetros, porém os mais comuns em redes neurais são Xavier e He.
4. A estratégia de hiperparâmetro a ser utilizada foi com experimentos em relação ao valor do "dropout", muito comum em redes neurais.
5. A combinação feita de hiperparâmetros para o "dropout" foi de 0, 0,1 e 0,2.
6. Foi feito o treinamento do modelo com o Código do Apêndice A.
7. Foram feitos testes com regularização L1 e L2, porém as mudanças foram muito baixas, logo, foi preferido utilizar unicamente o hiperparâmetro de "dropout".
8. Com os treinos com os três valores de "dropout", é escolhido o com o maior valor de "recall".

Após criar o modelo, ele deverá aparecer dentro do dataset em que foi criado, como mostrado na imagem abaixo:

Figura 44 – Criação do modelo no GCP

Fonte: Autoria própria.

Nele, pode-se clicar e visualizar os dados de treino e suas métricas ou executar o comando do Apêndice B, que com o resultado do código será possível realizar as próximas

ações. Com a avaliação do modelo, escolhe-se o ensaio com maior *recall* e assim realiza-se a predição do modelo para a geração de pontuações dos consumidores com o código descrito no Apêndice C, sendo colocadas em uma tabela. A partir dessa tabela, as duas últimas colunas são a probabilidade indicado por "prob" e o score que é o seu valor vezes mil. Agora que todos os consumidores possuem uma pontuação de 0 a 1000, é possível classificá-los com percentis.

4.3 Classificação dos consumidores com percentis

Utilizando o código do Apêndice D na tabela de resultado do código da predição, obtém-se 8 novas colunas em relação à anterior.

- previous_score: Pontuação na semana de referência anterior.
- current_score: Pontuação na semana de referência atual.
- current_tendency: Classificação na semana de referência atual.
- current_tendency_number: Classificação em número na semana de referência atual.
- previous_tendency: Classificação na semana de referência anterior.
- previous_tendency_number: Classificação em número na semana de referência anterior.
- tendency_status: Indicação da mudança de classificação da relação da semana de referência atual com a semana de referência anterior.
- score_status: Indicação da mudança de pontuação da relação da semana de referência atual com a semana de referência anterior.

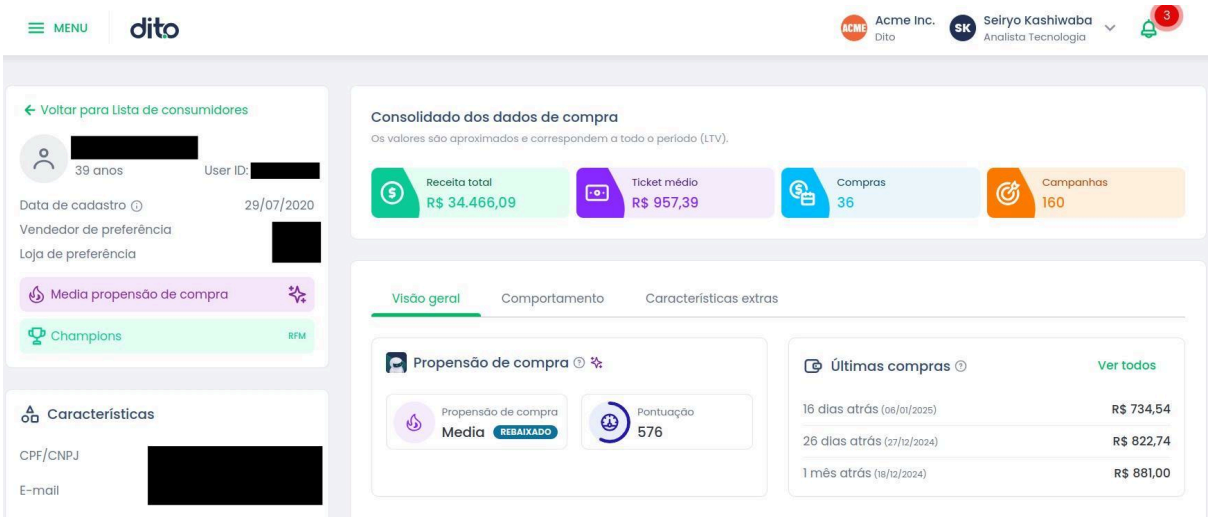
A coluna "current_tendency" irá mostrar a classificação atual do consumidor de acordo com a sua probabilidade, sendo as quatro classificações possíveis de "Alta Propensão", "Média Propensão", "Baixa Propensão" e "Sem Propensão". Após feita essa classificação, foram selecionados cerca de 4000 consumidores aleatoriamente para pertencerem ao grupo de controle e em seguida, foram selecionados mais cerca de 4000 consumidores que estavam classificados como "Alta Propensão" para compor o grupo de teste. Com esses dois grupos separados é possível realizar o teste A/B para verificar se a aplicação da inteligência artificial foi um sucesso.

4.4 Disparo das notificações para os consumidores em grupos de controle

Depois de feitas todas as classificações na tabela, foi feita a atualização das fichas dos consumidores para que possam ter a classificação de propensão para poder realizar o disparo

dos emails personalizados. A Figura 45 abaixo mostra um exemplo de um consumidor classificado como “Média Propensão” e com pontuação de 576.

Figura 45 – Dados do consumidor no aplicativo da Dito CRM



Fonte: Autoria própria.

Com as fichas atualizadas, foram criadas as campanhas e o segmento, como mostrado na subseção 3.7, uma com a classificação de “Alta Propensão” e também para consumidores sem classificação para a variante de controle.

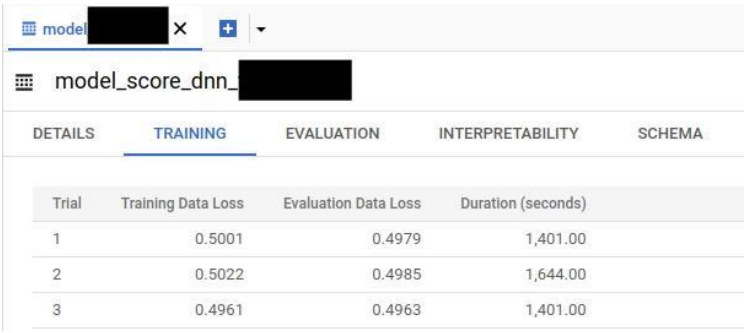
5 RESULTADOS

A subseção 5.1 será apresentado o resultado do treinamento do modelo da rede neural em que serão apresentadas as métricas de avaliação e a matriz de confusão dela. Na subseção 5.2 serão apresentados os resultados da criação das tabelas em que são geradas as propensões pelas predições e classificações dos consumidores pelo método estatístico de percentil. Na subseção 5.3 será apresentado o resultado comparativo entre o grupo de controle e o grupo de teste do teste A/B realizado.

5.1 Resultado do treinamento do modelo

Foram feitos três treinos variando o valor de "dropout" da rede neural. O BigQuery ML provê uma visualização para conseguir avaliar as métricas do modelo. A figura 46 abaixo apresenta a aba de treinamento em que se mostra os valores das perdas do modelo.

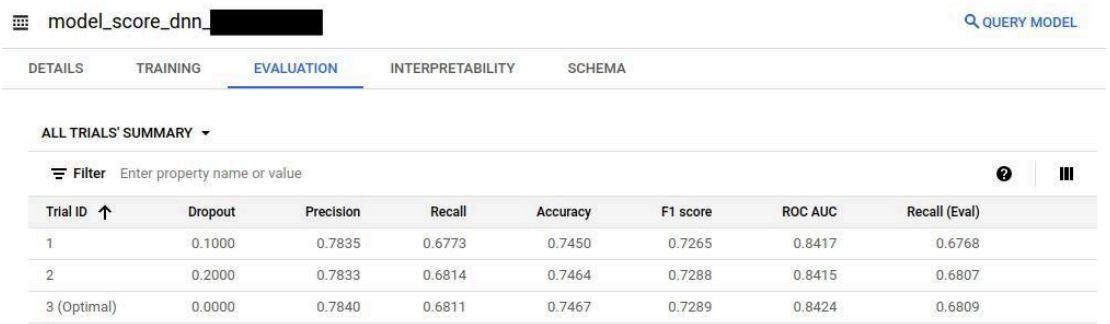
Figura 46 – Perdas do modelo e duração do treino no GCP



Fonte: Autoria própria.

Nota-se que ao tentar realizar alguma análise apenas com esses valores não será possível concluir a eficácia do modelo. A próxima aba de avaliação será possível realizar maiores análises sobre o modelo, apresentado na figura 47 abaixo.

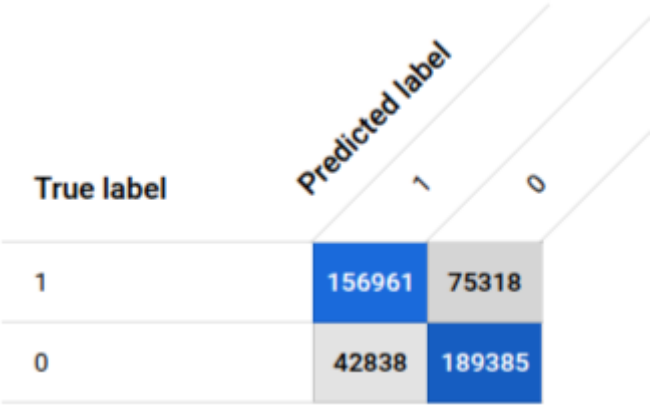
Figura 47 – Avaliação do modelo no GCP



Fonte: Autoria própria.

Com esses valores é possível analisar que o modelo sem a utilização de “dropout” obteve um maior sucesso se observar pela métrica de recall de avaliação apresentada na última coluna da imagem. Para aprofundar nesses números, a figura 48 abaixo apresenta a matriz de confusão do modelo, porém na tela, só é possível verificar os números com a linha de corte de 0,5064. Então se o consumidor está com 50,64% de chances de realizar uma compra, ele será considerado como propenso a comprar e abaixo dessa porcentagem será considerado como não propenso a comprar.

Figura 48 – Matriz de confusão do modelo no GCP



Fonte: Autoria própria.

Com essa matriz é possível analisar que:

- 156961 acertos na predições de consumidores que realmente realizaram uma compra.
- 42838 predições alegando que realizaram uma compra mas na verdade não realizaram.
- 75318 predições alegando que não realizaram compra mas na verdade realizaram.
- 189385 acertos nas predições de consumidores que realmente não realizaram uma compra.

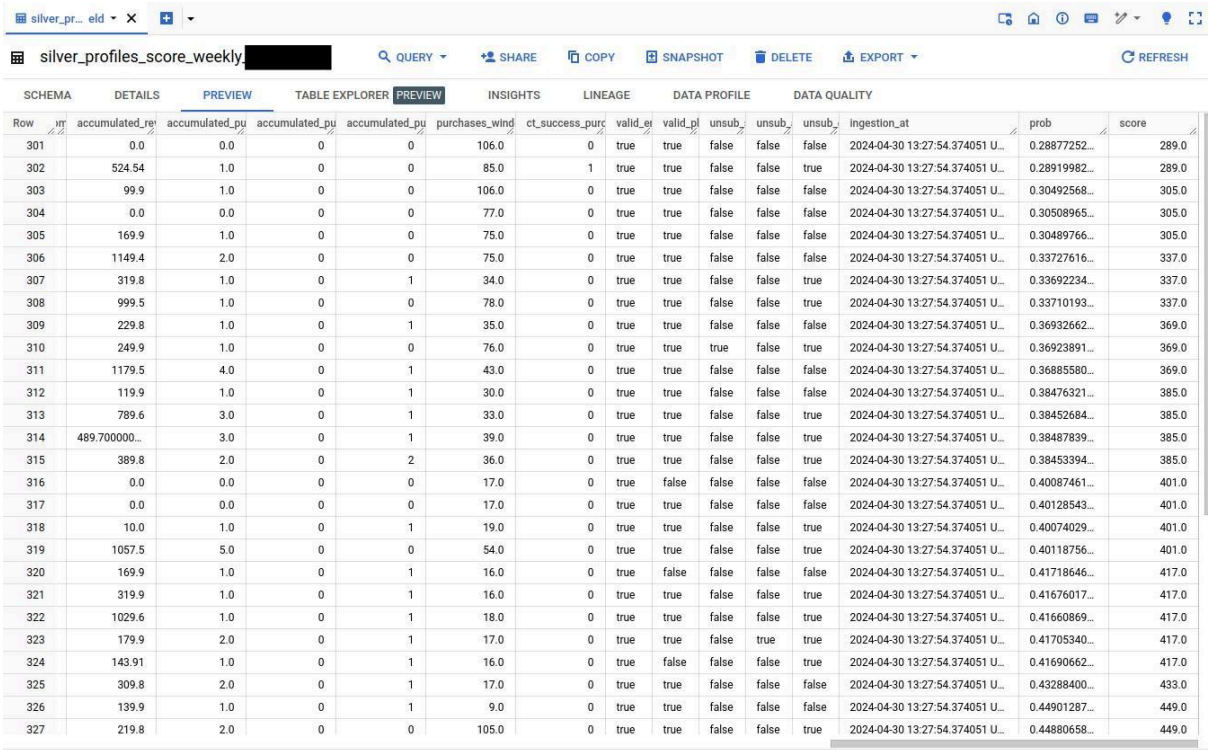
E como estamos medindo pela métrica de recall, então podemos concluir que de 232279 consumidores que realizaram compras, o modelo conseguiu acertar 156961, o que resulta no valor de 67,57%, em que se pode dizer que é um valor que pode ser considerado

bom, pois a cada 3 pessoas que realmente compraram, o modelo está conseguindo identificar duas delas.

5.2 Resultado das predições do modelo e classificações com percentil

A figura 49 abaixo apresenta pelas colunas “prob” e “score” a probabilidade do consumidor de realizar uma compra de acordo com a predição do modelo e seu *score* respectivamente. Vale ressaltar que o *score* é o valor da probabilidade vezes mil.

Figura 49 – Resultado da predição do modelo no GCP

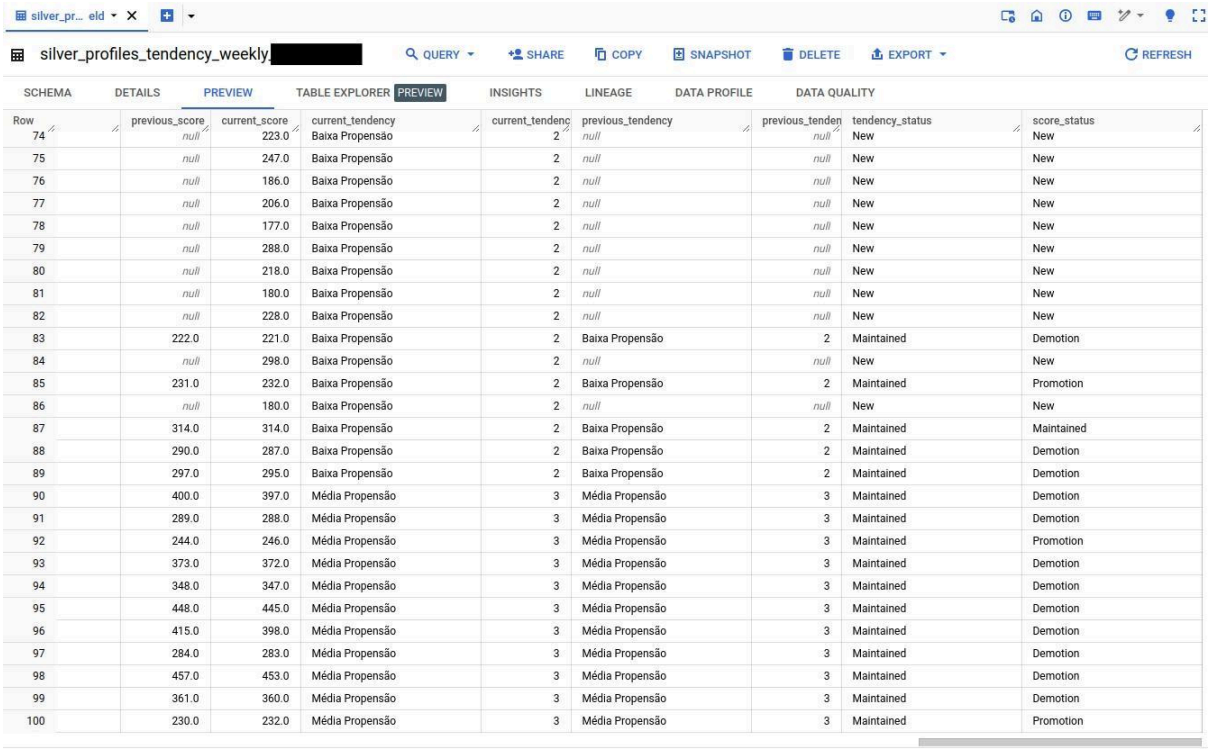


Row	accumulated_re	accumulated_pu	accumulated_pu	accumulated_pu	purchases_wind	ct_success_purc	valid_e	valid_pl	unsub_	unsub_	unsub_	ingestion_at	prob	score
301	0.0	0.0	0	0	106.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.28877252...	289.0
302	524.54	1.0	0	0	85.0	1	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.28919982...	289.0
303	99.9	1.0	0	0	106.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.30492568...	305.0
304	0.0	0.0	0	0	77.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.30508965...	305.0
305	169.9	1.0	0	0	75.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.30489766...	305.0
306	1149.4	2.0	0	0	75.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.33727616...	337.0
307	319.8	1.0	0	1	34.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.33692234...	337.0
308	999.5	1.0	0	0	78.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.33710193...	337.0
309	229.8	1.0	0	1	35.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.36932662...	369.0
310	249.9	1.0	0	0	76.0	0	true	true	true	false	true	2024-04-30 13:27:54.374051 U...	0.36923891...	369.0
311	1179.5	4.0	0	1	43.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.36885580...	369.0
312	119.9	1.0	0	1	30.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.38476321...	385.0
313	789.6	3.0	0	1	33.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.38452684...	385.0
314	489.700000...	3.0	0	1	39.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.38487839...	385.0
315	389.8	2.0	0	2	36.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.38453394...	385.0
316	0.0	0.0	0	0	17.0	0	true	false	false	false	false	2024-04-30 13:27:54.374051 U...	0.40087461...	401.0
317	0.0	0.0	0	0	17.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.40128543...	401.0
318	10.0	1.0	0	1	19.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.40074029...	401.0
319	1057.5	5.0	0	0	54.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.40118756...	401.0
320	169.9	1.0	0	1	16.0	0	true	false	false	false	false	2024-04-30 13:27:54.374051 U...	0.41718646...	417.0
321	319.9	1.0	0	1	16.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.41676017...	417.0
322	1029.6	1.0	0	1	18.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.41660869...	417.0
323	179.9	2.0	0	1	17.0	0	true	true	false	true	true	2024-04-30 13:27:54.374051 U...	0.41705340...	417.0
324	143.91	1.0	0	1	16.0	0	true	false	false	false	true	2024-04-30 13:27:54.374051 U...	0.41690662...	417.0
325	309.8	2.0	0	1	17.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.43288400...	433.0
326	139.9	1.0	0	1	9.0	0	true	true	false	false	false	2024-04-30 13:27:54.374051 U...	0.44901287...	449.0
327	219.8	2.0	0	0	105.0	0	true	true	false	false	true	2024-04-30 13:27:54.374051 U...	0.44880658...	449.0

Fonte: Autoria própria.

Com esses valores, é possível gerar as classificações de acordo com o seu respectivo percentil, a figura 50 apresenta como que se resultou essa classificação. A coluna “current_tendency” indica a classificação que o consumidor ficou. E com as classificações em cada consumidor, é possível realizar o experimento de teste A/B para poder realizar a comparação de desempenho dos grupos e concluir se a utilização de inteligência artificial realmente pode ser benéfica para a otimização de vendas.

Figura 50 – Classificação das predições do modelo no GCP



silver_profiles_tendency_weekly.[redacted]

QUERY | SHARE | COPY | SNAPSHOT | DELETE | EXPORT | REFRESH

SCHEMA | DETAILS | PREVIEW | TABLE EXPLORER PREVIEW | INSIGHTS | LINEAGE | DATA PROFILE | DATA QUALITY

Row	previous_score	current_score	current_tendency	current_tendenc	previous_tendency	previous_tenden	tendency_status	score_status
74	null	223.0	Baixa Propensão	2	null	null	New	New
75	null	247.0	Baixa Propensão	2	null	null	New	New
76	null	186.0	Baixa Propensão	2	null	null	New	New
77	null	206.0	Baixa Propensão	2	null	null	New	New
78	null	177.0	Baixa Propensão	2	null	null	New	New
79	null	288.0	Baixa Propensão	2	null	null	New	New
80	null	218.0	Baixa Propensão	2	null	null	New	New
81	null	180.0	Baixa Propensão	2	null	null	New	New
82	null	228.0	Baixa Propensão	2	null	null	New	New
83	222.0	221.0	Baixa Propensão	2	Baixa Propensão	2	Maintained	Demotion
84	null	298.0	Baixa Propensão	2	null	null	New	New
85	231.0	232.0	Baixa Propensão	2	Baixa Propensão	2	Maintained	Promotion
86	null	180.0	Baixa Propensão	2	null	null	New	New
87	314.0	314.0	Baixa Propensão	2	Baixa Propensão	2	Maintained	Maintained
88	290.0	287.0	Baixa Propensão	2	Baixa Propensão	2	Maintained	Demotion
89	297.0	295.0	Baixa Propensão	2	Baixa Propensão	2	Maintained	Demotion
90	400.0	397.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
91	289.0	288.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
92	244.0	246.0	Média Propensão	3	Média Propensão	3	Maintained	Promotion
93	373.0	372.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
94	348.0	347.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
95	448.0	445.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
96	415.0	398.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
97	284.0	283.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
98	457.0	453.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
99	361.0	360.0	Média Propensão	3	Média Propensão	3	Maintained	Demotion
100	230.0	232.0	Média Propensão	3	Média Propensão	3	Maintained	Promotion

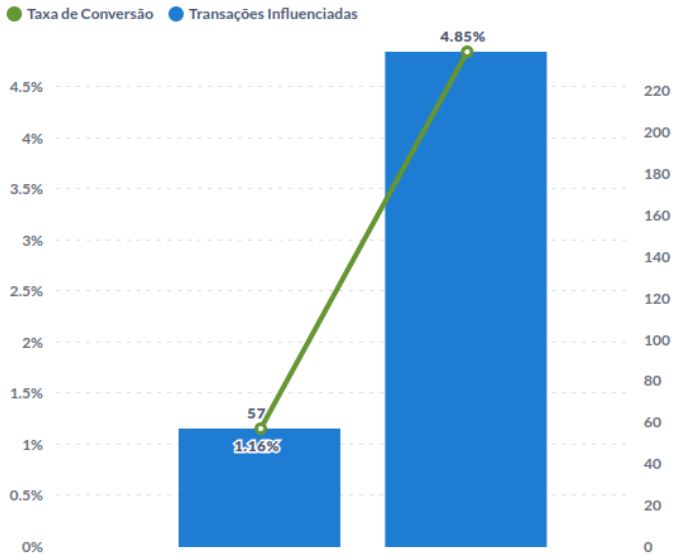
Fonte: Autoria própria.

5.3 Disparo das notificações para os consumidores

Para ilustrar o desempenho dos disparos, foram utilizados os gráficos de coluna e de inclinação vistos na subseção de visualização de dados, em que os dados de barra da esquerda indicam a variante de controle, ou seja, sem mudança na base de dados de consumidores, e os dados da barra direita indicam a variante de teste, ou seja, passaram por classificações utilizando as predições do modelo de redes neurais.

Em CRM, é muito comum utilizar a “transação influenciada” como métrica, pois ela mede a partir de um determinado tempo do disparo se houve uma compra. É muito comum as marcas utilizarem 7 dias de janela para medir essa conversão. E outra métrica muito utilizada é a taxa de conversão da campanha. Como mencionado anteriormente, cerca de 4000 consumidores foram selecionados para cada grupo e a figura 51 a seguir apresenta o resultado de acordos com as métricas de “transação influenciada” indicada pelo gráfico de colunas e taxa de conversão indicada pelo gráfico de inclinação.

Figura 51 – Transações e conversão das campanhas de SMS da variante de controle (à esquerda) e da variante de teste (à direita)



Fonte: Autoria própria

Analisando o gráfico anterior, nota-se que há uma variação de quase até 4 vezes mais de conversão ao utilizar inteligência artificial para auxiliar nas vendas da marca. Com isso pode-se dizer que as variáveis utilizadas para verificar se o consumidor está propenso ou não a realizar uma compra, está muito condizente com a realidade, pois esses disparos foram feitos com uma marca real de vestimentas esportivas.

6 CONCLUSÃO

Com o estudo realizado, pode-se dizer que a utilização da IA pode alavancar as vendas. Ela chegou para auxiliar em muitas áreas em que a compreensão humana não pode ser capaz. Nesse projeto, o procedimento experimental foi primeiramente realizado a coleta dos dados do comportamento dos consumidores como número de compras, total de receita gerada, número de cliques em produtos e entre outros, e a variável de saída sendo se o consumidor realizou uma compra ou não. Depois de coletar essas informações, é necessário realizar o tratamento desses dados, que é a parte de *feature engineering*. Depois de tratados os dados e modelados para a aplicação de ML, basta realizar o treinamento do modelo de redes neurais para adquirir a probabilidade do consumidor realizar uma próxima compra.

Com as probabilidades dos consumidores, foi aplicado o conceito estatístico de percentis para classificar os consumidores em quatro grupos: “Alta Propensão”, “Média Propensão”, “Baixa Propensão” e “Sem Propensão”. Com as classificações em mãos, foi feito o teste A/B para ver se a IA está realmente funcionando. Depois de separar em grupo de controle e de teste, foram feitos os mesmos disparos de campanhas para esses dois grupos com um software de CRM, porém um está com o uso da inteligência artificial, ou seja, com os consumidores mais prováveis de realizar uma compra.

A utilização do BigQuery, uma ferramenta robusta de análise de dados, foi essencial para o armazenamento e processamento eficiente das informações, viabilizando a execução de boa parte do projeto. Para concluir, foi visto que o grupo de teste teve uma melhora significativa na conversão de vendas de cerca de quatro vezes mais em relação ao grupo de controle. Então, é possível concluir que utilizar a IA corretamente, pode otimizar as vendas no setor varejista, potencializando os negócios e adquirir vantagem competitiva no mercado.

Como sugestões para oportunidades futuras, seria interessante encontrar alguma maneira de incorporar dados como gênero e localização geográfica que poderiam enriquecer ainda mais o modelo. Outro fator que poderia melhorar o modelo seria encontrar uma melhor técnica de classificar os consumidores, pois a maioria da base não realiza compras, então, muitas vezes classificada por posição de score, poderá remanejar os consumidores erroneamente. E por fim, deve-se atentar que todo esse teste foi realizado para uma única marca, então ainda existem inúmeros fatores que podem influenciar em más predições.

REFERÊNCIAS

AGGARWAL, Charu C. Neural networks and deep learning: a textbook. 2. ed. Cham: Springer, 2023.

BURKOV, A. Machine Learning Engineering. Publicado independente, 2020.

BUTTLE, F.; MAKLAN, S. Customer Relationship Management: Concepts and Technologies. 2. ed. Burlington: Butterworth-Heinemann, 2009.

EXAME. Uso de inteligência artificial é vantagem competitiva para empresas, diz MIT. Exame, 6 out. 2023. Disponível em: https://exame.com/future-of-money/uso-de-inteligencia-artificial-e-vantagem-competitiva-para-empresas-diz-mit/. Acesso em: 1 fev. 2025.

GEEKSFORGEEKS. Backpropagation in Neural Network. Disponível em: https://www.geeksforgeeks.org/backpropagation-in-neural-network/. Acesso em: 07 fev. 2025.

GOOGLE. BigQuery ML: Sintaxe para avaliar modelos. Google Cloud. Disponível em: https://cloud.google.com/bigquery/docs/reference/standard-sql/bigqueryml-syntax-evaluate. Acesso em: 9 fev. 2025.

GOOGLE. BigQuery ML: Sintaxe para criar modelos de redes neurais profundas. Google Cloud. Disponível em: https://cloud.google.com/bigquery/docs/reference/standard-sql/bigqueryml-syntax-create-dnn-models. Acesso em: 9 fev. 2025.

GOOGLE. BigQuery ML: Sintaxe para prever com modelos. Google Cloud. Disponível em: https://cloud.google.com/bigquery/docs/reference/standard-sql/bigqueryml-syntax-predict. Acesso em: 9 fev. 2025.

HELGEON, Lars. CRM for Dummies. 1. ed. New Jersey: For Dummies, 2017. 408 p. ISBN 978-1119368978.

KNAFLIC, Cole Nussbaumer**.** *Storytelling com dados: um guia sobre visualização de dados para profissionais de negócios.* Tradução de João Tortello. 2. ed. Rio de Janeiro: Alta Books, 2019.

LAKSHMANAN, Valliappa; TIGANI, Jordan. *Google BigQuery: the definitive guide: data warehousing, analytics, and machine learning at scale*. Sebastopol: O'Reilly Media, 2019.

NASSERY, Leemay**.** *Practical A/B Testing: Creating Experimentation-Driven Products.* Raleigh: Pragmatic Bookshelf, 2023.

ORACLE. Inteligência artificial para o varejo. Disponível em: https://www.oracle.com/br/retail/ai-retail/. Acesso em: 30 jan. 2025.

RUMSEY, Deborah J. *Statistics essentials for dummies*. Hoboken: Wiley Publishing, 2010.

SALESFORCE. Inteligência Artificial e CRM: como a IA está transformando o relacionamento com clientes. Disponível em: https://www.salesforce.com/br/blog/ia-e-crm/#h-tipos-de-ia-a-inteligencia-artificial-generativa. Acesso em: 30 jan. 2025.

SALESFORCE. O que é CRM? Entenda o significado e as vantagens para sua empresa. Disponível em: https://www.salesforce.com/br/blog/o-que-e-crm/. Acesso em: 30 jan. 2025.

THEOBALD, O. Machine Learning for Absolute Beginners: A Plain English Introduction. 2. ed. Publicado independente, 2017.

VEJA. Um ano de ChatGPT: Qual o futuro das inteligências artificiais? Disponível em: https://veja.abril.com.br/tecnologia/um-ano-de-chatgpt-qual-o-futuro-das-inteligencias-artificiais. Acesso em: 30 jan. 2025.

APÊNDICE A – Código da criação do modelo no BigQuery

```
      FOR BRAND_STRUCT IN (SELECT brand FROM
`project_id.feature_store.gold_feature_store_metrics`
                           WHERE score_model_metrics.recall IS NULL AND
total_unique_purchases <> 0)
         DO
         EXECUTE IMMEDIATE
         """CREATE OR REPLACE MODEL `dito-it-ml-prod.score.model_score_dnn_"""||
REPLACE(BRAND_STRUCT.brand,'-','_')||"""`
         OPTIONS(MODEL_TYPE = 'DNN_CLASSIFIER',
         AUTO_CLASS_WEIGHTS = TRUE,
         TF_VERSION = '2.8.0',
         OPTIMIZER = 'ADAGRAD',
         DROPOUT = HPARAM_CANDIDATES([0,0.1,0.2]),
         NUM_TRIALS = 3,
         MAX_PARALLEL_TRIALS = 3,
         HPARAM_TUNING_OBJECTIVES = ['RECALL']
         ) AS
         SELECT * except (version) FROM `project_id.score.raw_training_set_"""||
REPLACE(BRAND_STRUCT.brand,'-','_')||"""`;""";
      END FOR;
```

APÊNDICE B – Código da avaliação do modelo no BigQuery

```
SELECT
 *
FROM ML.EVALUATE(MODEL `project_id.score.model_score_dnn_X`)
```

APÊNDICE C – Código da predição do modelo no BigQuery

```
CREATE TEMP TABLE MODEL_DATA AS
WITH PREDICT AS
 (SELECT * FROM ML.PREDICT(MODEL `project_id.score.model_score_dnn_X`,
 (SELECT
   week_start_date,
   brand,
   user_id,
   agenda_influenced_purchases_before_contact,
   agenda_sum_influenced_revenue_before_contact,
   agenda_avg_window_hours_before_contact,
   agenda_contact_window_weeks,
   agenda_purchases_window_weeks,
   sms_influenced_purchases_before_contact,
   email_influenced_purchases_before_contact,
   sms_sum_influenced_revenue_before_contact,
   email_sum_influenced_revenue_before_contact,
   sms_avg_window_hours_before_contact,
   email_avg_window_hours_before_contact,
   sms_dispatch_window_weeks,
   email_dispatch_window_weeks,
   sms_purchases_window_weeks,
   email_purchases_window_weeks,
   cnt_open_notification_last_week,
   cnt_abandoned_cart_last_week,
   cnt_access_product_last_week,
   cnt_access_cart_last_week,
   cnt_access_home_last_week,
   cnt_open_notification_last_month,
   cnt_abandoned_cart_last_month,
   cnt_access_product_last_month,
   cnt_access_cart_last_month,
   cnt_access_home_last_month,
   accumulated_revenue,
   accumulated_purchases,
   accumulated_purchases_last_month,
   accumulated_purchases_last_year,
   purchases_window_weeks,
   IFNULL(ct_success_purchases,0) AS ct_success_purchases,
   valid_email_flag,
   valid_phone_flag,
   unsub_sms_flag,
   unsub_agenda_flag,
   unsub_email_flag,
   ingestion_at
   FROM `project_id.feature_store.gold_profiles_weekly`
   WHERE
     brand = 'dito-feras'
     AND ingestion_at > IFNULL((SELECT MAX(ingestion_at) FROM
`project_id.score.silver_profiles_score_weekly_X`),'1990-01-01')
     ))
```

```
  ),
SCORE_PROFILES AS
 (SELECT
   PREDICT.* except (predicted_label, predicted_label_probs, trial_id),
   pre.prob,
   round(pre.prob,3)*1000 as score
 from
   PREDICT, unnest(predicted_label_probs) as pre
 where
   pre.label = 1)
SELECT * FROM SCORE_PROFILES;

SELECT * FROM MODEL_DATA;
```

Apêndice D - Código da classificação com percentis no BigQuery

```
WITH RN_SCORE_DESC AS (
        SELECT
          *,
          ROW_NUMBER() OVER (PARTITION BY week_start_date ORDER BY score DESC)
rn_score
          FROM `project_id.score.silver_profiles_score_weekly_X`
        WHERE
              ingestion_at > IFNULL((SELECT MAX(ingestion_at) FROM
`project_id.score.silver_profiles_tendency_weekly_X`),'1990-01-01')
       ),
       PERCENTILES AS (
        SELECT DISTINCT
          week_start_date,
          CAST(PERCENTILE_CONT(rn_score, 0.25) OVER (PARTITION BY week_start_date)
AS INT) AS percentile_025,
          CAST(PERCENTILE_CONT(rn_score, 0.50) OVER (PARTITION BY week_start_date)
AS INT) AS percentile_050,
          CAST(PERCENTILE_CONT(rn_score, 0.75) OVER (PARTITION BY week_start_date)
AS INT) AS percentile_075,
          CAST(PERCENTILE_CONT(rn_score, 1.00) OVER (PARTITION BY week_start_date)
AS INT) AS percentile_100
        FROM RN_SCORE_DESC
       ),
       SCORE_VALUE AS (
        SELECT
          P.week_start_date,
          P.percentile,
          IFNULL(LAG(P.rn_score) OVER (PARTITION BY P.week_start_date ORDER BY
P.percentile ASC),0) rn_score_lag,
          P.rn_score,
          RSD.score,
          IFNULL(LAG(RSD.score) OVER (PARTITION BY P.week_start_date ORDER BY
P.percentile ASC),1000) score_lag
        FROM PERCENTILES UNPIVOT(rn_score FOR percentile IN
(percentile_025,percentile_050,percentile_075,percentile_100)) P
        RIGHT JOIN RN_SCORE_DESC RSD
          ON RSD.week_start_date = P.week_start_date
          AND RSD.rn_score = P.rn_score
        WHERE P.rn_score IS NOT NULL
       ),

       SILVER_PROFILES_TENDENCY AS (
        SELECT
        RSD.*,
        CONCAT(CAST(CAST(SV.score AS INT64) AS STRING),'-',CAST(CAST(SV.score_lag
AS INT64) AS STRING)) AS score_bound,
        CASE
          WHEN SV.percentile = 'percentile_025' THEN 4
          WHEN SV.percentile = 'percentile_050' THEN 3
          WHEN SV.percentile = 'percentile_075' THEN 2
```

```
          ELSE 1
        END AS tendency
       FROM RN_SCORE_DESC RSD
       JOIN SCORE_VALUE SV
        ON SV.week_start_date = RSD.week_start_date
        AND RSD.rn_score > SV.rn_score_lag
        AND RSD.rn_score <= SV.rn_score
       ),

       SILVER_PROFILES_TENDENCY_LAG AS (
        SELECT
          SPT.*,
          LAG(score) OVER (PARTITION BY brand, user_id ORDER BY SPT.week_start_date
ASC) previous_score,
          LAG(tendency) OVER (PARTITION BY brand, user_id ORDER BY
SPT.week_start_date ASC) previous_tendency,
        FROM SILVER_PROFILES_TENDENCY SPT
       )

       SELECT
        * except (tendency, previous_tendency, score, previous_score, rn_score),
        previous_score,
        score as current_score,
        CASE
          WHEN tendency = 1 THEN 'Sem Propensão'
          WHEN tendency = 2 THEN 'Baixa Propensão'
          WHEN tendency = 3 THEN 'Média Propensão'
          WHEN tendency = 4 THEN 'Alta Propensão'
        END AS current_tendency,
        tendency as current_tendency_number,
        CASE
          WHEN previous_tendency = 1 THEN 'Sem Propensão'
          WHEN previous_tendency = 2 THEN 'Baixa Propensão'
          WHEN previous_tendency = 3 THEN 'Média Propensão'
          WHEN previous_tendency = 4 THEN 'Alta Propensão'
        END AS previous_tendency,
        previous_tendency as previous_tendency_number,
        CASE
          WHEN previous_tendency > tendency THEN 'Demotion'
          WHEN previous_tendency < tendency THEN 'Promotion'
          WHEN previous_tendency = tendency THEN 'Maintained'
          ELSE 'New'
        END AS tendency_status,
        CASE
          WHEN previous_score > score THEN 'Demotion'
          WHEN previous_score < score THEN 'Promotion'
          WHEN previous_score = score THEN 'Maintained'
          ELSE 'New'
        END AS score_status
       FROM SILVER_PROFILES_TENDENCY_LAG;
```