Bruno Abtibol Ramos

Machine Learning para Previsão de Churn

PROJETO FINAL

DEPARTAMENTO DE INFORMÁTICA
Programa de Graduação em Engenharia da Computação

Rio de Janeiro
Julho de 2024

Pontifícia Universidade Católica
do Rio de Janeiro

Bruno Abtibol Ramos

Machine Learning para Previsão de Churn

Relatório de Projeto Final II

Relatório de Projeto Final, apresentado ao Programa de Engenharia da Computação, do Departamento de Informática da PUC-Rio como requisito parcial para a obtenção do titulo de Bacharel em Engenharia da Computação.

Orientador: Augusto Cesar Espíndola Baffa

Rio de Janeiro
Julho de 2024

Bruno Abtibol Ramos

Graduando em Engenharia da Computação na PUC - Rio

Ficha Catalográfica

Abtibol Ramos, Bruno

Machine Learning para Previsão de Churn / Bruno Abtibol Ramos; orientador: Augusto Cesar Espíndola Baffa. – 2024.

48 f: il. color. ; 30 cm

Projeto Final - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2024.

Inclui bibliografia

1. Aprendizado de Máquina. 2. Previsão de Churn. 3. Computação em nuvem. 4. Engenharia de Dados. 5. Finanças e Negócios. 6. Carteiras Digitais.
I. Baffa, Augusto. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Para meu pai e minha mãe, pelo apoio incondicional.

Agradecimentos

Para meus queridos pais, que sempre me apoiaram na minha caminhada acadêmica. Não sei o que seria sem vocês.

Para todos os amigos que fiz ao longo desses cinco recompensantes anos de graduação, que certamente levarei para a vida toda.

Para meu orientador Augusto Baffa, pela sua incessante disponibilidade e vontade de ajudar no projeto.

Para o time NG.CASH, que, através de sua excelência, me estimulou e continua estimulando a ser um profissional cada vez melhor.

Resumo

Abtibol Ramos, Bruno; Baffa, Augusto. **Machine Learning para Previsão de Churn**. Rio de Janeiro, 2024. 48p. Projeto Final – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Os modelos de Machine Learning têm se tornado cada vez mais presentes no mundo dos negócios. Em um mercado crescentemente competitivo, a previsão de churn - ou seja, o momento em que um usuário deixa de utilizar um produto ou serviço - tornou-se crucial para empresas que buscam aumentar a retenção de clientes. Este projeto tem como objetivo a criação de um modelo robusto de Machine Learning para prever o churn a nível empresarial. Utilizando a computação em nuvem, sistemas avançados de Engenharia de Dados, práticas recomendadas de Aprendizado de Máquina e estratégias efetivas de alavancagem de negócios, o projeto espera fornecer uma ferramenta eficiente e escalável para prever churn em um banco digital, podendo servir de base para a construção de muitos outros modelos e também contribuir para a implementação de modelos de Machine Learning em empresas.

Palavras-chave

Aprendizado de Máquina; Previsão de Churn; Computação em nuvem; Engenharia de Dados; Finanças e Negócios; Carteiras Digitais;.

Abstract

Abtibol Ramos, Bruno; Baffa, Augusto (Advisor). **Machine Learning for Churn Prediction**. Rio de Janeiro, 2024. 48p. Projeto Final – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Machine Learning models have become increasingly present in the business world. In an increasingly competitive market, churn prediction — that is, the moment when a user stops using a product or service — has become crucial for companies seeking to increase customer retention. This project aims to create a robust Machine Learning model to predict churn at an enterprise level. Utilizing cloud computing, advanced Data Engineering systems, good Machine Learning practices, and effective business leverage strategies, the project hopes to provide an efficient and scalable tool to predict churn in a digital bank. This model can serve as a basis for building many other models and also contribute to the implementation of Machine Learning models in companies.

Keywords

Machine Learning; Churn Prediction; Cloud Computing; Data Engineering; Finance and Business; Digital Wallets.

Sumário

Lista de figuras

Lista de tabelas

"Practice isn't the thing you do once you're good. It's the thing you do that makes you good."

Malcolm Gladwell, *Outliers: The Story of Success.*

1
Introdução

Nos últimos anos, o crescimento de dados nas empresas transformou radicalmente a forma com a qual elas interagem com seus clientes. Um entendimento maior sobre o consumidor final é, em grande parte, resultado de um fortalecimento da cultura orientada a dados, seja na coleta de mais informações, melhores práticas de engenharia ou tomadas de decisão *data-driven* (CHKONIYA, 2020). Assim, estudar o comportamento do usuário tornou-se mais fácil com a maior disponibilidade de dados.

Nesse contexto, entender o consumidor final através de dados tornou-se prática quase obrigatória para empresas novas, tendo em vista um mundo cada vez mais competitivo (AFFAIRS, 2011). Compreender o seu perfil, o que o leva a tomar certas decisões, o que ele procura e como retê-lo são algumas das perguntas que podem ser respondidas através dessa ferramenta. Dessa forma, o estudo do churn, ou seja, quantos clientes deixam de usar o seu produto, se tornou mais factível para empresas orientadas a dados e possivelmente as fornece imensa vantagem competitiva - costuma ser entre 5-6 vezes mais caro adquirir um cliente novo do que um cliente já adquirido (MARKETING, 2010).

A previsibilidade do churn é particularmente importante em setores como telecomunicações, serviços financeiros, SaaS (Software as a Service) e varejo, onde a concorrência é intensa e a lealdade do cliente é difícil de manter. Uma empresa de telecomunicações, por exemplo, pode usar modelos de previsão de churn para identificar clientes em risco de cancelamento e oferecer promoções personalizadas para mantê-los. Da mesma forma, um banco digital pode utilizar a previsão de churn para desenvolver estratégias de engajamento que aumentem a fidelidade dos seus usuários.

Assim, muitas companhias estão construindo métodos mais complexos para previsão de churn através de Aprendizado de Máquina (NASIR, 2018). O objetivo desses modelos é tentar identificar a probabilidade de certo usuário deixar de consumir o produto e, assim, elaborar estratégias de retenção e *Customer Relationship Management* (CRM) para não perdê-lo. A aplicação do Aprendizado de Máquina em previsões de churn envolve o uso de algoritmos que analisam padrões históricos de comportamento do usuário e identificam sinais de desengajamento.

Este trabalho foi estruturado da seguinte maneira:

– **Capítulo 1: Introdução** - Apresenta a motivação e contexto do estudo, destacando a importância do uso de dados para entender e prever o

comportamento do consumidor.

- **Capítulo 2: Situação Atual** - Discute o estado atual do nicho de mercado e revisa trabalhos relevantes relacionados ao problema de previsão de churn.
- **Capítulo 3: Objetivos do Trabalho** - Define os objetivos principais do projeto, incluindo as metas a serem alcançadas e os requisitos funcionais da ferramenta desenvolvida.
- **Capítulo 4: Pesquisas Realizadas** - Aborda as tecnologias de nuvem em aplicações de negócios, MLOps na Amazon Web Services, o desenvolvimento do modelo preditivo e a avaliação dos modelos de Machine Learning.
- **Capítulo 5: Projeto e Especificação do Sistema** - Detalha os serviços utilizados e a análise das bases de dados originais.
- **Capítulo 6: Implementação e Avaliação** - Descreve a infraestrutura final implementada e os resultados obtidos.
- **Capítulo 7: Considerações Finais** - Detalha as contribuições e desafios do projeto e propõe melhorias para o futuro.
- **Capítulo 8: Referências Bibliográficas** - Lista todas as fontes e referências utilizadas ao longo do trabalho.

A motivação para este estudo está enraizada na necessidade crescente de empresas modernas se adaptarem a um ambiente de negócios altamente competitivo e orientado por dados. Este projeto visa fornecer um guia abrangente de como implementar um modelo robusto de Machine Learning a nível empresarial, utilizando computação em nuvem, sistemas avançados de Engenharia de Dados, boas práticas de Aprendizado de Máquina e técnicas reais de alavancagem de negócio.

Este estudo aplicará essas tecnologias ao caso da NG.CASH, uma carteira digital inovadora voltada para a Geração-Z, demonstrando na prática como tais abordagens podem ser implementadas para gerar valor real e mensurável para a empresa. A análise dos dados da NG.CASH permitirá identificar padrões de comportamento específicos dessa demografia, contribuindo para a formulação de estratégias de retenção eficazes e personalizadas para jovens usuários.

Em resumo, a previsão de churn utilizando Aprendizado de Máquina não só representa um avanço tecnológico importante, mas também uma oportunidade para empresas inovadoras como a NG.CASH se destacarem no mercado, oferecendo um serviço mais adaptado e eficiente aos seus clientes. ‘

2
Situação Atual

Os diferentes métodos de previsão de churn têm sempre o mesmo objetivo final: aumentar a retenção. Segundo uma pesquisa da Bain & Company (REICHHELD, 2011), um aumento de 5% na taxa de retenção em empresas financeiras pode resultar em um crescimento de 25% na receita. Sendo assim, o alcance desses resultados pode se dar por inúmeros métodos e combinações diferentes de técnicas de Machine Learning. Alguns autores utilizaram *Support Vector Machines* (SVM), Redes Neurais Artificiais (ANN) e algoritmos de *Random Forests*, enquanto outros optaram por focar mais no pré-processamento dos dados através do balanceamento de amostras e engenharia de features (V; K, 2016).

Inúmeras pesquisas comprovam que o Aprendizado de Máquina gera resultados impressionantes para a previsão do churn. Os modelos aprendem através da análise de dados passados e características particulares dos usuários. Gavril apresentou um método para estimar o churn baseado nos dados de 3333 usuários e 21 características, dividindo os clientes em duas classes (Yes/No) (A.K. et al., 2019). Utilizando *Principal Component Analysis* (PCA) para reduzir a dimensionalidade, o autor utilizou alguns dos algoritmos supracitados para chegar em uma *Area Under Curve* (AUC) de mais de 99% para todas as abordagens.

No caso de instituições financeiras como a NG.CASH, os profissionais de dados conseguem obter uma quantidade expressiva de informações do perfil do usuário através de seu comportamento transacional. É possível estimar a sua renda, seus gostos, seus hábitos e muitas outras características apenas observando como, quanto e onde gasta. Essa gama relevante de fatores torna a predição do churn mais robusta. Apesar disso, as empresas do setor apresentam uma crescente dificuldade de manter ou aumentar a retenção de seus clientes (MOSTAFA, 2020), uma vez que abrir contas em outras instituições é cada vez mais fácil e rápido.

Assim, conclui-se que o desafio de prever churn é imprescindível para o sucesso das companhias no contexto atual. O cenário mais competitivo torna essa prática uma necessidade, enquanto o maior acesso a dados também a torna mais acessível.

3
Objetivos do Trabalho

Neste projeto, serão exploradas diferentes abordagens de Aprendizado de Máquina e como combiná-las para obter resultados mais satisfatórios. Valhendo-se do ecossistema de dados da NG.CASH, o estudo buscará adequar, tratar e validar os dados através de pipelines de Engenharia de Dados; modelar e testar abordagens diferentes de *Machine Learning*; e concluir e validar os resultados alcançados.

O objetivo final do projeto é servir de inspiração para a introdução de modelos de Machine Learning em produção em outras empresas. O trabalho implementará uma infraestrutura robusta em nuvem (AWS) que permitirá plugar de forma eficaz novos modelos. Essa infraestrutura será pensada em termos de eficiência, robustez, escalabilidade e custo.

A variável resposta será a probabilidade de churn. O modelo de classificação será binário, tentando classificar usuários entre duas classes: não darão churn (0) e darão churn (1).

1. Para a criação da infraestrutura em nuvem, utilizar-se-á o ecossistema da AWS e seus serviços. Os recursos da AWS utilizados serão: Step Functions, Lambda Functions, Simple Storage System (S3), EventBridge, EMR Serverless e Amazon SageMaker. Para a manipulação e pré-processamento dos dados, utilizaremos o *framework* de *Big Data* chamado **Spark**, executado na plataforma de clusters auto-gerenciáveis **Elastic Map Reduce** (Amazon EMR). O armazenamento dos dados será feito no serviço de Datalake da AWS **Simple Storage Service** (S3).

2. As linguagens de programação utilizadas serão o Python (3.9) e o Typescript (Nodejs 18). A definição da infraestrutura em nuvem será feita através da biblioteca de IaaC AWS Cloud Development Kit (CDK), que permite criar recursos em nuvem de forma eficiente.

3. Para a construção e treinamento do modelo, será escolhida a biblioteca **sci-kit learn** do *Python*, que apresenta uma ampla gama de algoritmos de classificação e recursos de pré-processamento. O treinamento será feito no **Amazon SageMaker**, ferramenta de aprendizado automático da AWS.

4. Toda a orquestração da pipeline de treinamento será feita através dos recursos da AWS **EventBridge** e **Step Functions**.

Após o desenvolvimento do código e da infraestrutura necessária para o treinamento do modelo, o projeto buscará analisar a eficácia do mesmo através de diferentes técnicas como a AUC, Curva ROC, F1-Score etc. Feito isso, será elaborada uma estratégia de negócios integrada aos resultados do modelo.

4
Pesquisas Realizadas

Neste capítulo serão detalhadas as pesquisas realizadas para a produção desta tese. Dentre elas, há estudos relacionados a negócios, infraestrutura de nuvem, algoritmos de Machine Leaning e outros.

4.1
Tecnologias de Nuvem em Aplicações de Negócios

No dia 6 de agosto de 2012, o robô da NASA Curiosity pousou na superfície de Marte em uma missão exploratória. Além de ter sido um importante marco para o mundo científico, o episódio também representou um grande acontecimento para o mundo da computação em nuvem: grande parte infraestrutura tecnológica da missão foi construída em cima de recursos da Amazon Web Services para o streaming ao vivo e armazenamento de imagens. (JOURNALS, 2024)

O impacto do desenvolvimento das tecnologias de nuvem e seus recursos direcionados às práticas de Aprendizado de Máquina tem transformado a produção de negócios e pesquisas. O que antes era apenas disponibilizado para governos e multinacionais, passou a ser acessível para qualquer um com um computador e um cartão de crédito. A maior disponibilidade de poder e armazenamento computacional barato tem democratizado a possibilidade da criação de análises profundas de dados. Além disso, a adoção dessas tecnologias por empresas reduz, em grande escala, os esforços para a manutenção da infraestrutura computacional dedicada ao desenvolvimento dos modelos (GARCIA et al., 2020).

Nesse contexto, grandes companhias como a Meta, Netflix, Airbnb desfrutam dos serviços da Amazon Web Services para armazenar dados e treinar modelos (SINHA, 2020), enquanto outras organizações como Rolls-Royce, Heineken e a própria Microsoft utilizam a Microsoft Azure para tal. Outra opção bastante explorada no mundo empresarial é o Google Cloud, presente na infraestrutura da Uber, Spotify e Youtube (MARR, 2016).

Os principais provedores de nuvem fornecem serviços de IaaS (Infrastructure as a Service), SaaS (Software as a Service) e PaaS (Platform as a Service), nos quais existem dezenas de recursos criados para atender inúmeras necessidades. Afinal, a escolha da melhor infraestrutura em nuvem depende de diversos fatores como preço, sinergia de serviços, escalabilidade e outros motivos particulares a cada empresa (DUTTA; DUTTA; XORIANT, 2019).

A fim de implementar essas tecnologias, principalmente no que diz respeito à arquitetura de Machine Learning, companhias têm desenvolvido áreas de MLOps (Machine Learning Operations) que trabalham com serviços em nuvem para produzir modelos e objetivam auxiliar a integração contínua e a implantação rápida e repetitiva de novos projetos de aprendizagem.

4.2 MLOps na Amazon Web Services

No contexto da NG.CASH, a empresa hospeda seus servidores e armazena dados na infraestrutura da AWS, além de possuir créditos que reduzem o valor do pagamento mensal dessa utilização. Assim, torna-se óbvia a escolha pelos recursos da AWS na implantação da arquitetura de ML. Portanto, a seguir serão explorados alguns dos principais recursos da AWS no universo de Aprendizagem de Máquina.

4.2.1 Elastic MapReduce (EMR)

O Elastic Map Reduce é a solução da Amazon para disponibilizar frameworks de Big Data, como o Hadoop e o Spark, sendo executado nos datacenters da empresa. Esse recurso permite que organizações foquem no processamento massivo de dados sem se preocupar com o gerenciamento de recursos, possuindo um modelo de pay-as-you-go. É, portanto, uma ferramenta poderosa que possibilita que empresas iniciem projetos de Machine Learning com baixo custo e escalabilidade instantânea. (SCHMIDT, 2013)

O EMR performa utilizando o framework MapReduce. Esse framework divide os dados de entrada em fragmentos reduzidos que são distribuídos entre os nós que compõem o cluster. A topologia atual do Amazon EMR agrupa suas instâncias em 3 grupos lógicos de instâncias: grupo mestre, que executa o YARN Resource Manager e o serviço HDFS Name Node; grupo principal, que executa o HDFS DataNode Daemon e o serviço YARN Node Manager, e grupo de tarefas, que executa o serviço YARN Node Manager. Além disso, utiliza como serviço padrão de armazenamento o S3 (Simple Storage System), que será detalhado mais adiante.

O recurso é disponibilizado através de três formas: EMR on EC2, EMR on EKS e EMR Serverless. Enquanto o primeiro se integra com o Elastic Compute Cloud (EC2), plataforma de clusters de processamento da Amazon, o EMR on EKS (Elastic Kubernete Service) permite que usuários criem containers de aplicações para executar os programas de processamento em massa. O EMR Serverless é considerado o mais adequado para os casos em que o usuário não

possui uma boa previsibilidade do tamanho da base a ser processada, além de ser o mais fácil de gerenciar - uma vez que não é necessário se preocupar com as configurações de clusters e nós.

4.2.2 Simple Storage Service (S3)

O Amazon S3 (Simple Storage Service) é um serviço de armazenamento de arquivos oferecido pela AWS com vasta escalabilidade, segurança e performance. Este serviço permite que usuários armazenem e busquem qualquer tipo de dado de forma eficiente, sendo amplamente utilizado em uma gama relevante de casos de uso, como websites, aplicações móveis, backup e restauração, arquivamento, dispositivos IoT, análises de big data e Machine Learning.

O S3 armazena dados como objetos dentro de buckets (baldes). Um objeto consiste em um arquivo e opcionalmente qualquer metadado que descreva esse arquivo. Cada bucket pode conter múltiplas pastas para facilitar a organização. Existem diferentes classes de armazenamento no S3, como o S3 Standard para uso geral, o S3 Intelligent-Tiering para dados com padrões de acesso variáveis, o S3 Glacier para arquivamento de longo prazo, e outros, cada um oferecendo diferentes níveis de disponibilidade, durabilidade e custos.

Muitas pessoas ainda possuem certa desconfiança no que diz respeito à segurança de dados em nuvem. Contudo, o S3 fornece controles abrangentes para manter os dados seguros. A Amazon utiliza o framework IAM (Identity and Access Management) para gerenciar usuários, autenticar e autorizar acessos (SAEED SARAH BARAS, 2019). Contas de usuário e grupos possuem diferentes permissões definidas por políticas de acesso, garantindo que cada usuário tenha somente os privilégios necessários.

No S3, as políticas de acesso podem ser definidas por recurso ou por usuário, permitindo flexibilidade no gerenciamento. A integridade e a segurança dos dados são mantidas com recursos como versionamento, replicação e criptografia SSL/TLS para proteger comunicações entre cliente e console. A integridade é verificada usando códigos de autenticação, assinaturas digitais e criptografia. A autenticação multifator (MFA), tanto virtual quanto por hardware, adiciona uma camada extra de proteção, minimizando os riscos de acesso não autorizado. (SAEED SARAH BARAS, 2019).

4.2.3 Step Functions

As Step Functions da AWS é um recurso projetado para orquestrar diferentes tipos de workflows. Dentre os casos de uso, tem-se a automação de

processos de ETL (Extract, Transform and Load), orquestração de processos paralelizados em larga escala, microsserviços e serviços de segurança e TI (MATHEW VASILIOS ANDRIKOPOULOS, 2021). Diversas empresas utilizam este serviço em seus processos internos, como por exemplo o Taco Bell e a Nationwide Children's.

Este produto da nuvem está relacionado ao escopo de FaaS (Function-as-a-Service), portanto, é uma forma de computação Serverless em que o usuário não precisa se preocupar com a construção da complexa infraestrutura por trás do lançamento de microsserviços e outros tipos de workflows (GROGAN et al., 2020).

A AWS, pioneira no oferecimento de recursos de FaaS, também possui outros recursos que se conectam com as Step Functions de forma intuitiva, como a AWS Lambda, explicada na seção a seguir. A junção desses recursos permite que sejam criadas aplicações mais complexas, com máquinas de estados condicionais (MATHEW VASILIOS ANDRIKOPOULOS, 2021). Além disso, as Step Functions possuem a capacidade de ordenar a execução de outros recursos da AWS com facilidade, o que será importante para a interconexão de todos os serviços associados ao pré-processamento, treinamento, teste e deploy do modelo de Machine Learning do projeto deste trabalho.

4.2.4 AWS Lambda

AWS Lambda é outro serviço de computação serverless oferecido pela Amazon Web Services. Ele permite a execução de código em resposta a eventos sem a necessidade de provisionar ou gerenciar servidores. Portanto, desenvolvedores precisam se preocupar apenas com o código e as lógicas de negócio enquanto a AWS cuida do gerenciamento da infraestrutura por trás (SBARSKI, 2017). Este recurso é utilizado em diversos tipos de aplicação, como processamento de dados, backends de aplicações web e móveis e até orquestração de microsserviços.

Sendo um dos serviços mais utilizados da AWS, as funções Lambda são consideradas extremamente escaláveis e altamente eficientes, podendo aumentar ou diminuir a quantidade de instâncias em execução conforme a demanda. Quando um evento dispara uma função Lambda, o serviço automaticamente aloca recursos computacionais para executar o código. Após a execução, esses recursos são desalocados, garantindo que o usuário pague apenas pelo tempo de computação utilizado (SBARSKI, 2017).

4.2.5 AWS SageMaker

O AWS SageMaker é um serviço de aprendizado de máquina totalmente gerenciado pela Amazon Web Services (AWS). Ele facilita a criação, treinamento e implantação de modelos de aprendizado de máquina em escala. O SageMaker é utilizado em várias áreas como análise preditiva, processamento de imagens e visão computacional, processamento de linguagem natural e serviços de recomendação personalizada. (JOSHI, 2019)

O produto fornece um ambiente de desenvolvimento integrado (IDE) com Jupyter Notebooks, onde os usuários podem realizar a preparação dos dados, o desenvolvimento e o treinamento dos modelos. Ele oferece uma variedade de algoritmos pré-configurados e otimizados, bem como a capacidade de importar bibliotecas personalizadas. Além disso, permite que sejam implantados modelos como endpoints HTTPs para inferência em tempo real ou em lote utilizando dados armazenados no S3. (MISHRA, 2019)

Dessa forma, o SageMaker oferece um ambiente completo para o ciclo de vida do Machine Learning. Contudo, como será explicado com mais detalhamento na seção da implementação do projeto, escolheu-se realizar algumas etapas do processo utilizando outros recursos que não o SageMaker, como o EMR Serverless, Lambda Functions e Step Functions.

4.3 Algoritmos para o Desenvolvimento do Modelo

O modelo de Machine Learning no projeto deste trabalho pode ser treinado através de algoritmos de classificação binária, uma vez que a variável resposta quer responder se o usuário vai ou não dar churn. Dentre eles, escolheu-se focar no estudo e implementação de 5 algoritmos: Naïve Bayes, Redes Neurais Artificiais, Regressão Logística, Support Vector Machines e Árvore de Decisão.

Nesta seção serão aprofundados todos os algoritmos implantados no desenvolvimento do modelo.

4.3.1 Naïve Bayes

O Naïve Bayes é um algoritmo de classificação probabilístico baseado no Teorema de Bayes. A lógica Bayesiana é uma abordagem lógica para atualizar a probabilidade de uma hipótese a partir de novas evidências, tendo um papel crucial na ciência (BERRAR, 2001).

Os classificadores Naïve Bayes são consideravelmente escaláveis, sendo avaliados através de uma fórmula fechada de complexidade linear, portanto, possuindo uma performance consideravelmente melhor que outros algoritmos de classificação que utilizam métodos iterativos.

4.3.1.1 Teorema de Bayes

O Teorema de Bayes é a base do algoritmo Naïve Bayes e é expresso da seguinte forma:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{4-1}$$

onde:

- $P(A|B)$ é a probabilidade a posteriori (probabilidade condicionada) de A condicional a B.
- $P(B|A)$ é a probabilidade a posteriori (probabilidade condicionada) de B condicional a A.
- $P(A)$ é a probabilidade a priori do evento A.
- $P(B)$ é a probabilidade a priori do evento B.

4.3.1.2 Classificador Naïve Bayes

Para classificar uma nova instância, o algoritmo Naïve Bayes calcula a probabilidade posterior para cada classe. A classe com a maior probabilidade posterior é escolhida como a classe predita (RISH*, 2019). O classificador Naïve Bayes pode ser expresso como:

$$\hat{y} = \arg\max_{y_j} \prod_{k=1}^{p} P(x_k|y_j)P(y_j) \tag{4-2}$$

onde:

- $\hat{y}$ é a classe predita.
- y_k é a k-ésima classe.
- x_i é a i-ésima feature.
- p é o número de features.

4.3.1.3 Supondo Independência

A principal suposição do Naïve Bayes é que as features são condicionalmente independentes, dado a classe. Isso simplifica o cálculo das probabilidades conjuntas, pois podemos escrever:

$$P(x|y) = \prod_{i=1}^{n} P(x_i|y) \tag{4-3}$$

Essa suposição de independência, embora rara na prática, permite que o Naïve Bayes seja muito eficiente e ainda forneça bons resultados em muitos problemas reais. Além disso, o cálculo das probabilidades condicionais é bastante paralelizável e, portanto, passível de ser realizada através de frameworks de processamento distribuído como o MapReduce.

4.3.2 Redes Neurais Artificiais

As Redes Neurais Artificiais representam uma variedade de algoritmos e sistemas que tentam replicar o funcionamento de neurônios nos sistemas nervosos centrais dos seres humanos. Essencialmente, são compostas por nós agrupados que recebem, processam e transmitem sinais (PANTIC, 2023). Um neurônio pode receber um ou vários sinais e, através de uma ponderação desses sinais através de pesos e funções de ativação, produz um resultado que pode vir a ser a entrada de outro neurônio na sequência.

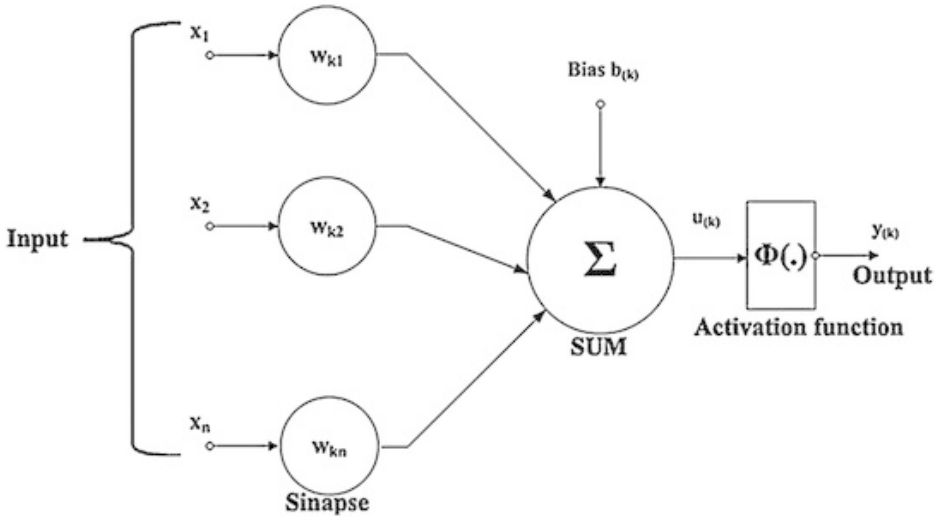

Figura 4.1: Neurônio Matemático

Como representado na imagem 4.1, os sinais de entrada (x_1,x_2, ..., x_n) são recebidos pelo neurônio e multiplicados por pesos (w_1,w_2, ..., w_n), que representam a importância que uma determinada entrada irá ter na saída

do neurônio. Os pesos, inicialmente, são definidos aleatoriamente e vão se ajustando durante o decorrer do treinamento da rede (KROGH, 2008).

Após a multiplicação, o neurônio soma cada par x^*w através de função de soma "$\sum$". Normalmente, esta etapa é representada por um produto escalar de um vetor de entradas X por um vetor de pesos W, e, depois, é ajustada através do Viés (*Bias*), representado como $b_{(k)}$ na imagem.

O resultado após o ajuste do Viés é passado por uma função de ativação $\Phi(.)$ A função de ativação em redes neurais artificiais introduz não-linearidade ao modelo, permitindo que a rede aprenda e represente relações complexas entre os dados de entrada e saída. Sem essa não-linearidade, a rede seria equivalente a um modelo linear, incapaz de capturar padrões complexos. As funções mais utilizadas são: Sigmóide, ReLU, Tangente Hiperbólica e Softmax (ABDOLRASOL, 2021).

Por fim, o valor produzido pela função de ativação se torna a saída do neurônio. Como explicado anteriormente, essa saída pode ser passada para outros neurônios, em sequência, ou até mesmo ser a saída definitiva da rede.

O objetivo do treinamento de uma rede neural é ajustar os pesos e vieses dos neurônios para que a rede aprenda a mapear as entradas para as saídas desejadas de maneira precisa. Isso é feito minimizando a função de perda, que mede a diferença entre as previsões da rede e os valores reais. Técnicas como o gradiente descendente e a retro-propagação (*backpropagation*) são usadas para atualizar iterativamente os pesos e vieses, melhorando gradualmente o desempenho da rede até que ela se torne capaz de generalizar bem os dados de treinamento (KROGH, 2008).

4.3.3 Regressão Logística

A Regressão Logística é um método estatístico amplamente utilizado para problemas de classificação binária (HARRIS, 2021). Diferentemente da regressão linear, que é utilizada para prever valores contínuos, a regressão logística pode ser usada para prever a probabilidade de uma variável dependente binária, assumindo valores 0 ou 1.

4.3.3.1 Modelo de Regressão Logística

Quando a variável dependente Y é binária, seu comportamento segue a distribuição de Bernoulli (GONZALEZ, 2018). Portanto, tem-se que a probabilidade de sucesso é 0 <= p <= 1 e a probabilidade de fracasso é q = 1 - p.

O modelo logístico objetiva estimar p para uma combinação linear de variáveis independentes (SHENG et al., 2022). A partir da inversa da função logit, ou função sigmoide, chega-se na equação principal do modelo:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{4-4}$$

onde z é uma combinação linear das features de entrada. Neste contexto, z é definido como:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \tag{4-5}$$

onde, β_0 é o intercepto e $\beta_1, \beta_2, \ldots, \beta_n$ são os coeficientes das features $x_1, x_2, \ldots, x_n$, respectivamente. A probabilidade de a variável dependente ser 1, dado um conjunto de features de entrada, é então dada por:

$$P(y = 1|x) = \sigma(z) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n)}} \tag{4-6}$$

4.3.3.2 Treinamento do Modelo

O treinamento de um modelo de Regressão Logística envolve a estimação dos parâmetros $\beta_0, \beta_1, \ldots, \beta_n$ que maximizam a verossimilhança dos dados observados (NASRI, 2024). A função de custo mais comumente utilizada é a função de log-verossimilhança, definida como:

$$L(\beta) = \sum_{i=1}^{m} [y_i \log(\sigma(z_i)) + (1 - y_i) \log(1 - \sigma(z_i))] \tag{4-7}$$

onde m é o número de observações no conjunto de treinamento. A otimização dessa função pode ser realizada usando algoritmos como gradiente descendente.

4.3.3.3 Aplicações e Vantagens

A Regressão Logística é amplamente utilizada em várias áreas, incluindo medicina (para prever a presença de doenças), marketing (para prever a resposta do cliente a campanhas), e finanças (para prever inadimplência de crédito) (MARTINS, 2023). Suas principais vantagens incluem:

- Simplicidade e interpretabilidade dos resultados.
- Eficiência em termos de tempo de computação e memória.
- Capacidade de fornecer probabilidades associadas às previsões.

4.3.4 Support Vector Machines

O Support Vector Machine (SVM) é um algoritmo de aprendizado supervisionado amplamente utilizado para tarefas de classificação binária. O principal objetivo do SVM é encontrar um hiperplano que melhor separe os dados em duas classes distintas, maximizando a margem entre os pontos de dados mais próximos de cada classe (conhecidos como vetores de suporte) (CERVANTES et al., 2020).

4.3.4.1 Definição Formal

Dado um conjunto de treinamento $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, onde $\mathbf{x}_i \in \mathbb{R}^d$ são os vetores de características e $y_i \in \{-1, 1\}$ são os rótulos das classes, o SVM resolve o seguinte problema de otimização:

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{sujeito a} \quad y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n, \end{aligned}$$

onde $\mathbf{w}$ é o vetor de pesos, b é o termo de bias, ξ_i são as variáveis de folga que permitem que alguns pontos de dados sejam classificados incorretamente, e C é um hiperparâmetro que controla a penalidade para erros de classificação.

4.3.4.2 Kernel Trick

Para lidar com problemas não linearmente separáveis, o SVM pode ser estendido para o SVM com Kernel (SHEYKHMOUSA, 2020). A ideia é mapear os dados de entrada para um espaço de características de maior dimensão, onde um hiperplano linear pode separar as classes. Este mapeamento é realizado implicitamente através de uma função de kernel $K(\mathbf{x}_i, \mathbf{x}_j)$, que calcula o produto interno no espaço de características:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j).$$

Alguns dos kernels mais comuns incluem o kernel linear, o kernel polinomial e o kernel RBF (Radial Basis Function).

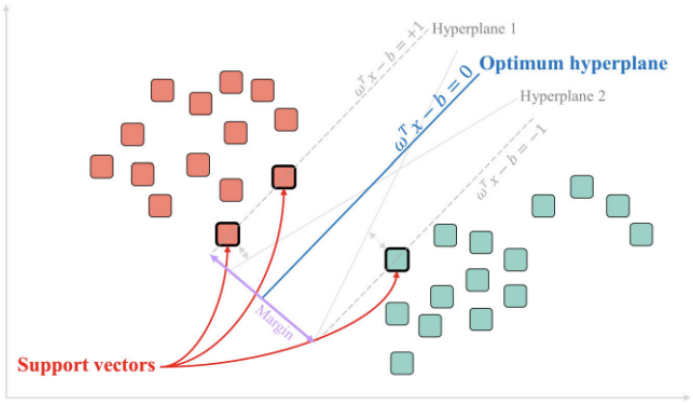

Figura 4.2: SVM em ação com dados linearmente separáveis

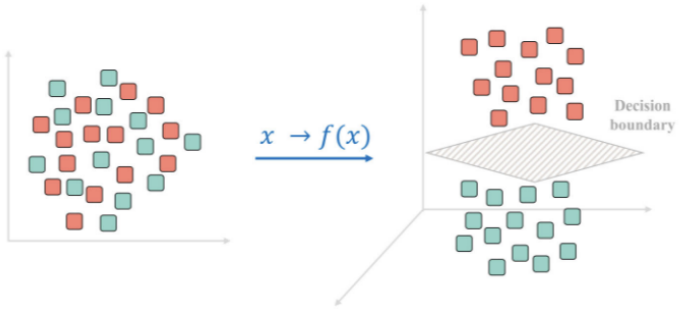

Figura 4.3: SVM com dados não linearmente separáveis (Kernel Trick)

4.3.5 Árvores de Decisão

Árvores de decisão são um método popular para tarefas de classificação binária devido à sua interpretabilidade e simplicidade. Uma árvore de decisão classifica exemplos ao fazer uma série de perguntas baseadas nos atributos dos dados, resultando em uma estrutura de árvore onde cada nó representa uma decisão sobre um atributo e cada folha representa uma classe (JUNIOR, 2016).

4.3.5.1 Construção da Árvore

A construção de uma árvore de decisão envolve a escolha do melhor atributo para dividir os dados em cada nó. Este processo é repetido recursivamente até que todos os dados sejam classificados ou outro critério de parada seja atingido (SAFAVIAN; LANDGREBE, 1991). A escolha do atributo é geralmente baseada em métricas como o *gini impurity* ou o ganho de informação (*information gain*).

Dado um conjunto de dados D, o *gini impurity* de um nó é dado por:

$$G(D) = 1 - \sum_{i=1}^{C} p_i^2,$$

onde p_i é a proporção de exemplos da classe i no conjunto de dados D, e C é o número de classes. O objetivo é minimizar o *gini impurity* ao escolher o atributo para divisão.

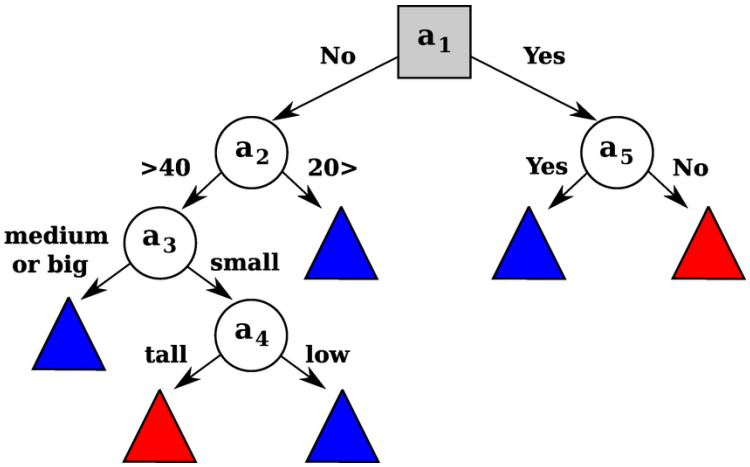

Figura 4.4: Exemplo de uma árvore de decisão

4.3.5.2 Poda da Árvore

Para evitar o sobreajuste (*overfitting*), pode-se aplicar técnicas de poda (*pruning*) à árvore de decisão (SAFAVIAN; LANDGREBE, 1991). A poda remove ramos da árvore que têm pouca importância para a classificação, baseando-se em um conjunto de validação ou em um critério de complexidade.

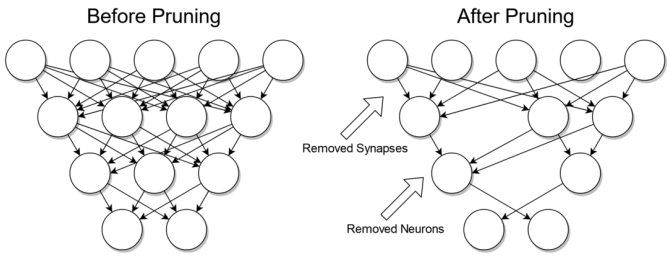

Figura 4.5: Árvore de decisão podada

4.4 Avaliação de Modelos de Machine Learning

Avaliar a eficiência de um modelo de Machine Learning é crucial para garantir que ele tenha um bom desempenho e possa generalizar bem para dados não vistos. A seguir, serão apresentadas as principais métricas e métodos utilizados para a avaliação de modelos de classificação binária (HANDELMA et al., 2018).

4.4.1 Métricas de Avaliação

4.4.1.1 Acurácia

A acurácia mede a proporção de previsões corretas em relação ao total de previsões realizadas:

$$\text{Acurácia} = \frac{\text{Número de previsões corretas}}{\text{Total de previsões}}$$

4.4.1.2 Precisão

A precisão indica a proporção de verdadeiros positivos (VP) entre todas as previsões positivas (VP + FP):

$$\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}}$$

4.4.1.3 Recall (Sensibilidade ou Revocação)

O recall mede a proporção de verdadeiros positivos entre todos os casos reais positivos (VP + FN):

$$\text{Recall} = \frac{\text{VP}}{\text{VP} + \text{FN}}$$

4.4.1.4 F1-Score

O F1-Score é a média harmônica da precisão e do recall, sendo útil quando há um desbalanceamento entre as classes:

$$\text{F1-Score} = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}}$$

4.4.1.5
Matriz de Confusão

A matriz de confusão apresenta os números de verdadeiros positivos (VP), falsos positivos (FP), verdadeiros negativos (VN) e falsos negativos (FN):

		Previsões	
		Positivo	Negativo
Valores Reais	Positivo	VP	FN
	Negativo	FP	VN

4.4.1.6
Área sob a Curva ROC (AUC-ROC)

A curva ROC (Receiver Operating Characteristic) é um gráfico que mostra a relação entre a taxa de verdadeiros positivos (TPR) e a taxa de falsos positivos (FPR) (RAINIO JARMO TEUHO, 2024). A AUC (Area Under the Curve) representa a capacidade do modelo em distinguir entre as classes.

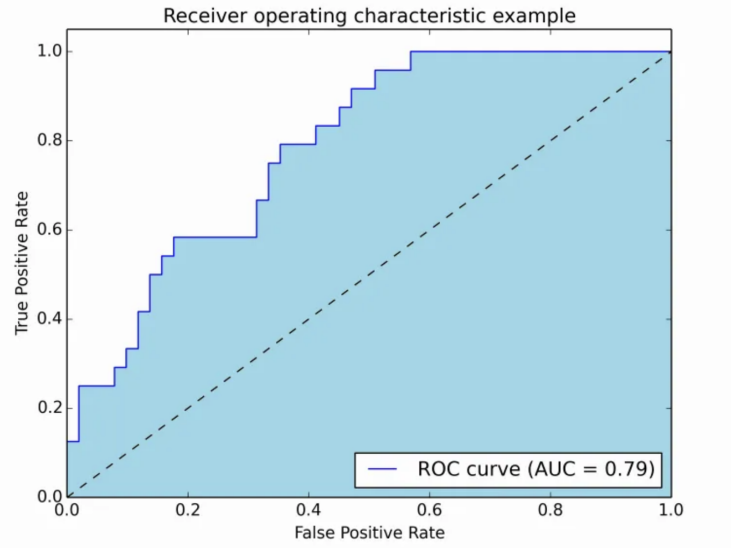

Figura 4.6: Exemplo de ROC/AUC

4.4.2
Métodos de Avaliação

4.4.2.1
Validação Cruzada (Cross-Validation)

A validação cruzada divide os dados em k subconjuntos (folds). O modelo é treinado k vezes, cada vez usando $k-1$ folds para treinamento e 1 fold para

teste (HANDELMA et al., 2018). A média das métricas de avaliação é usada para estimar a performance do modelo.

4.4.2.2 Conjunto de Validação (Validation Set)

Separa os dados em três conjuntos: treino, validação e teste. O conjunto de validação é usado para ajustar os hiperparâmetros do modelo.

4.4.2.3 Teste A/B (A/B Testing)

Utilizado em ambientes de produção, onde duas variantes do modelo são comparadas para ver qual tem melhor desempenho.

5
Projeto e Especificação do Sistema

Neste capítulo, serão abordadas mais informações sobre os serviços e bibliotecas necessários para a implementação do projeto. Ademais, será realizada uma descrição sobre as bases originais do modelo, que serão depois processadas e juntadas em um único dataset de features relevantes sobre cada usuário, detalhada posteriormente.

5.1
Serviços

Alguns pré-requisitos deverão ser atendidos para a construção do projeto:

1. **Conta AWS**: Uma conta na AWS para utilizar os serviços da AWS SageMaker, EMR Serverless, Step Functions e outros serviços relacionados.
2. **Configuração da AWS CLI**: Instalar e configurar a AWS CLI (Command Line Interface) com as credenciais da sua conta AWS.
3. **Instalação do Node.js e npm**: Instalar o Node.js e npm (Node Package Manager) para utilizar o AWS CDK com TypeScript.
4. **Instalação do AWS CDK**: Instalar o AWS CDK globalmente usando npm.
5. **Configuração do Ambiente de Desenvolvimento**: Certificar-se de ter Python instalado e criar um ambiente virtual para isolar as dependências do projeto.
6. **Instalação das Dependências do CDK**: Instalar as dependências necessárias para trabalhar com o CDK em TypeScript.
7. **Conjunto de Dados**: Preparar o conjunto de dados que será utilizado para treinar o modelo, armazenando-o em um bucket S3 acessível pelo SageMaker e pelo EMR.
8. **Conta no GitLab**: Uma conta no GitLab para gerenciar o código-fonte, a colaboração no projeto e a rotina de CI/CD.

5.2 Análise das Bases Originais

O dataset final, ou seja, o conjunto de dados no qual serão aplicados os algoritmos detalhados no capítulo anterior, é composto por uma combinação de bases analíticas já existentes no ecossistema de dados da NG.CASH. Dentre elas, destacam-se as bases transacionais, bases cadastrais (informações sobre o usuário e seu processo de criação de conta) e bases de cartões. Destas bases internas, é possível extrair diversas informações interessantes para a análise dos modelos, como a atividade transacional do usuário quebrada em diferentes produtos (criptoativos, transações bandeiradas, PIX), informações de logradouro e idade do usuário, além de atributos binários como se o usuário tem mesada cadastrada, se tem cartão, se tem outra conta bancária etc.

A partir dessas bases, o script em PySpark será responsável por processá-las e montar uma base final de features de usuários. A seguir, será detalhado o dataset final e seus atributos.

5.2.1 Churn Dataset

O dataset final, chamado de Churn Dataset, possui atributos originados a partir da junção das bases explicadas na seção anterior. São eles:

- `account_id`: Identificador único da conta de cada usuário. Este atributo é utilizado para diferenciar cada registro de usuário no dataset. Esse identificador é numérico e foi anonimizado para a pesquisa.
- `age_today`: Idade atual do usuário. Este atributo é calculado com base na data de nascimento do usuário e pode influenciar no comportamento de churn.
- `residential_state`: Estado de residência do usuário. Este atributo pode ser relevante para identificar padrões geográficos de churn.
- `months_since_acc_creation`: Número de meses desde a criação da conta. Este atributo indica o tempo de relacionamento do usuário com o banco.
- `transaction_count`: Número total de transações realizadas pelo usuário. Este atributo pode refletir o nível de atividade e engajamento do usuário.
- `days_since_last_transaction`: Número de dias desde a última transação realizada pelo usuário. Este atributo ajuda a identificar a recência da atividade do usuário.

- `has_virtual_card`: Indicador binário (sim/não) se o usuário possui um cartão virtual. Este atributo pode influenciar o comportamento de uso e, consequentemente, o churn.
- `has_plastic_card`: Indicador binário (sim/não) se o usuário possui um cartão físico. Similar ao cartão virtual, este atributo pode impactar o comportamento de churn.
- `average_balance`: Saldo médio mantido pelo usuário em sua conta. Este atributo pode ser um indicador da saúde financeira e do engajamento do usuário com o banco.
- `crypto_ttv`: Valor total transacionado em criptomoedas pelo usuário. Este atributo pode indicar o interesse do usuário em produtos financeiros inovadores.
- `card_ttv`: Valor total transacionado usando cartões (físico e virtual) pelo usuário. Este atributo é um indicador do uso dos cartões e pode estar relacionado ao churn.
- `pix_ttv`: Valor total transacionado usando o sistema de pagamentos instantâneos (PIX). Este atributo pode mostrar o engajamento do usuário com meios de pagamento modernos.
- `has_allowance`: Indicador binário (sim/não) se o usuário possui um pagamento recorrente (mesada, por exemplo). Este atributo pode indicar que haverá movimentação na conta do usuário independentemente da utilização ativa dela.
- `has_other_account`: Indicador binário (sim/não) se o usuário possui contas em outras instituições financeiras. Este atributo pode afetar a probabilidade de churn, indicando se seria mais fácil para o usuário trocar de conta bancária.
- `will_churn`: Variável resposta indicando se o usuário irá churnar (1) ou não (0). Este atributo é a variável dependente que o modelo de Machine Learning está tentando prever.

6
Implementação e Avaliação

Neste capítulo serão detalhadas todas as partes essenciais para a implementação do modelo em produção, como a arquitetura dos recursos de nuvem, o racional por trás do pré-processamento, a construção e comparação dos algoritmos de predição e a análise dos resultados obtidos.

O repositório público disponibilizado pode ser acessado através do seguinte link: **https://github.com/brunoabtramos/customer-churn-ml**.

O leitor poderá acompanhar a explicação e se basear na estrutura do repositório para implementar seu próprio modelo de ML utilizando AWS. O repositório contém a infraestrutura completa em CDK e exemplos de scripts de pré-processamento, treinamento, envio de eventos e funções operacionais. No entanto, os scripts específicos utilizados para a confecção do modelo deste projeto, por serem de caráter sensível e particular à NG.CASH, não serão disponibilizados integralmente.

6.1
Infraestrutura Final

A infraestrutura final do modelo e seu Workflow são apresentados na Figura abaixo. A seguir, é detalhada cada etapa do processo:

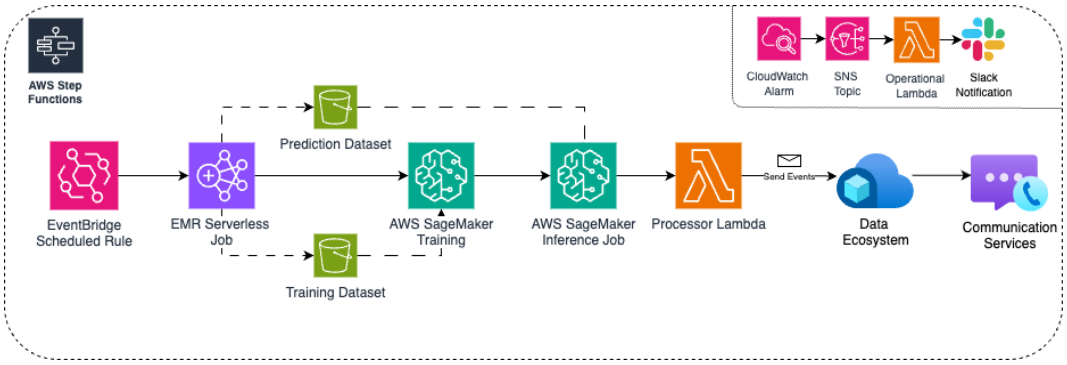

Figura 6.1: Arquitetura dos recursos em nuvem utilizados

6.1.1
Workflow

O processo se inicia com uma regra agendada do Amazon EventBridge, que dispara periodicamente a execução de uma Step Function. Esta Step Function coordena a execução das etapas subsequentes do Workflow, orquestrando os trabalhos no EMR Serverless, SageMaker e funções Lambda.

Primeiramente, a Step Function aciona um trabalho no EMR Serverless, responsável por pré-processar os dados, tanto de treino quanto de predição. Os dados de treino são utilizados para treinar o modelo, enquanto os dados de predição são utilizados posteriormente para inferência. A utilização do EMR nesta etapa é bastante adequada, uma vez que o grande volume de dados e a complexidade das transformações exige um poder de processamento massivo e paralelizável.

A base de treinamento é construída a partir de um recorte das bases da NG.CASH de 3 meses para trás. Isso porque, segundo a definição de churn para grande parte dos bancos digitais, um usuário terá dado churn apenas quando não tiver transacionado por 3 meses. Assim, todas as variáveis independentes serão calculadas a partir das ações dos usuários neste período anterior a 3 meses, enquanto a variável resposta (will_churn) é modelada a partir da verificação da presença de cada usuário na base transacional nos últimos 3 meses. Já a base de predição é montada a partir das bases atuais e, assim, a variável resposta será predita pelos parâmetros do treinamento.

Após o pré-processamento, os dados de treino são passados para um trabalho de treinamento no AWS SageMaker, onde um modelo de Aprendizado de Máquina é treinado. Os cinco algoritmos aprofundados no Capítulo 4 serão testados e o melhor deles (a partir do cálculo da AUC-ROC) será efetivamente aplicado na base de predição. O script de treinamento construído seleciona automaticamente o algoritmo de melhor desempenho e o salva em um arquivo .joblib no SageMaker para ser posteriormente aplicado à base real.

Com o modelo treinado, a próxima etapa envolve um trabalho de inferência no AWS SageMaker, que aplica o modelo treinado aos dados de predição para prever se os usuários irão ou não dar churn.

Os resultados da inferência são então enviados para uma função AWS Lambda de processamento. Esta função processa os resultados e envia os eventos correspondentes para o ecossistema de dados, que por sua vez será responsável por enviá-los para os serviços de CRM que se comunicam diretamente com os usuários que possuem alta probabilidade de churn. Estes serviços utilizam diversas plataformas para notificação, incluindo e-mail, SMS e aplicativos de mensagens.

Neste projeto, contudo, não será aprofundada a etapa de comunicação com o cliente, uma vez que é um processo consideravelmente particular da NG.CASH. O leitor deverá apenas se basear na infraestrutura construída até o envio dos eventos para o ecossistema de dados.

6.2 Estrutura do repositório

O repositório segue uma estrutura organizada que facilita o desenvolvimento, teste e implantação do projeto. Na raiz do repositório, se encontram arquivos de configuração essenciais como o `.gitignore`, que especifica quais arquivos e diretórios devem ser ignorados pelo Git, e o `.gitlab-ci.yml`, que configura o GitLab CI/CD para automação de integração e deploy contínuo - explicado na seção seguinte. Também estão presentes o `Dockerfile`, usado para construir imagens Docker que podem ser necessárias para a execução no EMR Serverless, e arquivos de configuração de dependências e compilação, como `package.json`, `package-lock.json`, e `tsconfig.json`.

A pasta `bin` contém scripts de inicialização ou configuração de entrada para a aplicação. A pasta `cdk.out` é gerada pelo AWS Cloud Development Kit (CDK) durante a síntese e contém os artefatos resultantes, como templates do CloudFormation. Já a pasta `lib` é onde reside o código-fonte principal da aplicação, organizado em subdiretórios conforme sua funcionalidade específica. Dentro da pasta `lib`, encontramos a pasta `applications/handlers`, que é subdividida em `event-processor`, `ml-processor`, `operational`, e `pre-processor`, cada uma contendo scripts específicos para diferentes partes do processamento e treinamento do modelo.

Na pasta `ml-processor`, temos o script em Python `training-script.py`, que é responsável pelo treinamento do modelo de Machine Learning. A pasta `pre-processor` contém o script `pre-processor.py`, essencial para a etapa de pré-processamento dos dados. A estrutura também inclui uma pasta `constructors` - hoje inutilizada, mas que poderia conter abstrações dos construtores da infraestrutura principal contida no arquivo `customer-churn-ml-stack.ts`.

A organização clara e modular do repositório facilita a colaboração e manutenção do projeto, permitindo que diferentes componentes sejam desenvolvidos e testados de forma independente antes de serem integrados. Esta abordagem modular, juntamente com a automação proporcionada pelo CI/CD do GitLab, garante que o desenvolvimento seja ágil, seguro e eficiente.

- **CUSTOMER-CHURN-ML/**
  - bin/

    customer-churn-ml.ts Arquivo padrão de criação da aplicação do CDK.
  - lib/

applications/handlers/ Contém as pastas de todos os scripts e funções.

event-processor/ Pasta contendo a função de geração e envio de eventos.

handler.ts Arquivo que contém a função de geração e envio de eventos.

ml-processor/ Pasta que contém o script de treinamento.

training-script.py Script de treinamento.

operational/ Scripts operacionais.

alarms.types.ts Tipos de alarmes para o handler operacional.

operational-lambda.handler.ts Lambda operacional que dispara uma mensagem no Slack se a Step Function falhar.

pre-processor/ Pasta que contém o script de pré-processamento.

pre-processor.py Script de pré-processamento.

constructors/ Construtores do stack (inutilizado).

customer-churn-ml-stack.ts Arquivo que define a Stack principal.

- node_modules/ Módulos Node.js.
- test/ Pasta para testes unitários.
- .env Arquivo de configuração de ambiente.
- .gitignore Arquivo para ignorar arquivos no Git.
- .gitlab-ci.yml Configuração do GitLab CI/CD.
- .npmignore Arquivo para ignorar arquivos no npm.
- cdk.json Configuração do CDK.
- Dockerfile Arquivo Dockerfile.
- jest.config.js Configuração do Jest.
- package-lock.json Arquivo de bloqueio do npm.
- package.json Arquivo de configuração do npm.
- README.md Arquivo README.
- tsconfig.json Configuração do TypeScript.

6.3 Deploy em produção - CI/CD

O processo de deploy da infraestrutura em produção utilizando CI/CD no GitLab é configurado através do arquivo `.gitlab-ci.yml`, que define os estágios e jobs a serem executados. O pipeline é composto por três estágios principais: setup, deploy e pós-deploy. No estágio de setup, são instaladas ferramentas necessárias como o AWS CLI. No estágio de deploy, o código é construído e implantado utilizando o AWS CDK, além de incluir as cópias dos scripts de pré-processamento e treinamento para o S3. O estágio de pós-deploy realiza ações como a fusão de branches para manter a consistência entre os ambientes.

A configuração do CI/CD no GitLab permite a automação do processo de deploy, garantindo a entrega contínua e segura das mudanças. Com a definição clara dos estágios e jobs no arquivo `.gitlab-ci.yml`, é possível orquestrar de forma eficiente todas as etapas necessárias para o deployment da infraestrutura em produção. Essa prática não só melhora a eficiência do processo de desenvolvimento, mas também reduz os riscos associados ao deploy manual, promovendo um ciclo de desenvolvimento ágil e confiável.

6.4 Avaliação dos Resultados

Para verificar o sucesso dos resultados, deve-se avaliar tanto o funcionamento da infraestrutura quanto a qualidade do modelo gerado pelo trabalho de treinamento.

6.4.1 Configurações dos Recursos em Nuvem e Parâmetros de Treinamento

Para este projeto, como anteriormente explicado, escolheu-se utilizar o EMR Serverless em detrimento do EMR on EC2 e EMR on EKS, devido à sua capacidade de executar tarefas de Spark de forma escalável e sem a necessidade de gerenciar clusters manualmente. O código de pré-processamento, escrito em Python, é armazenado no S3 e referenciado na execução do trabalho EMR. Este design permite flexibilidade e escalabilidade, aproveitando a eficiência do Spark para lidar com grandes volumes de dados de forma eficiente. A execução do trabalho é monitorada continuamente, verificando o estado do trabalho até sua conclusão ou falha.

O trabalho de treinamento do SageMaker permite a seleção de instâncias otimizadas para Machine Learning. Para o projeto, escolheu-se a instância ml.m5.xlarge, que além de ser eficiente é relativamente barata. Além disso,

para executar o arquivo de treinamento, foi necessário desenvolver uma imagem Docker personalizada contendo o script de treinamento e as dependências necessárias, definidas em um Dockerfile. Essa imagem foi construída localmente e depois enviada para um repositório do Amazon Elastic Container Registry (ECR). É importante destacar que o SageMaker oferece vários algoritmos integrados que podem ser utilizados para treinamento de modelos de Machine Learning sem a necessidade de escrever um código de treinamento, contudo, para este projeto, escolheu-se utilizar código pois isso o torna mais flexível para diferentes abordagens.

Para otimizar os modelos e buscar melhores resultados, foram aplicadas algumas técnicas como a normalização dos valores numéricos, a substituição de valores vazios por moda e mediana e também o ajuste de hiperparâmetros através da função `GridSearchCV` do `sci-kit learn`.

Haja vista a alta sensibilidade dos dados transacionais e cadastrais dos usuários da NG.CASH, os dados foram anonimizados e treinados em apenas uma amostra da base de usuários para fins de pesquisa e avaliação. O modelo no qual serão analisados os resultados possui apenas uma amostra de 10.000 usuários ativos. Esta amostragem foi feita a partir da distribuição igualitária entre 10 faixas de percentis relativos ao número de transações (`transaction_count`). Já o modelo em produção será treinado a partir dos dados de todos os usuários ativos na foto das bases de 3 meses para trás, como explicado anteriormente.

6.4.2 Ajuste de hiperparâmetros

O ajuste de hiperparâmetros, também conhecido como *hyperparameter tuning*, é um processo essencial em Machine Learning que envolve a seleção de um conjunto ótimo de hiperparâmetros para um algoritmo de aprendizado. Diferente dos parâmetros aprendidos a partir dos dados durante o treinamento, os hiperparâmetros são definidos antes do processo de treinamento e controlam o comportamento do algoritmo.

A escolha adequada de hiperparâmetros pode melhorar significativamente o desempenho de um modelo, aumentando a precisão, reduzindo o overfitting e melhorando a capacidade de generalização para novos dados. Técnicas como *GridSearchCV* e *RandomizedSearchCV* são frequentemente utilizadas para explorar diferentes combinações de hiperparâmetros de maneira sistemática, permitindo a identificação das configurações que oferecem os melhores resultados para um dado conjunto de dados e problema específico. Este processo é vital para otimizar modelos complexos e garantir que eles forneçam

previsões robustas e confiáveis. A seguir, serão detalhados os hiperparâmetros testados no script de treinamento para cada algoritmo.

O Naïve Bayes é um modelo que não possui muitos hiperparâmetros ajustáveis, assim utilizou-se a configuração padrão, sem parâmetros adicionais.

Para o modelo de Rede Neural Artificial, vários hiperparâmetros foram escolhidos a fim de explorar diferentes arquiteturas e comportamentos da rede. O parâmetro *hidden_layer_sizes* define a configuração da arquitetura da rede neural. Foram testadas três configurações: (50,50,50) com três camadas escondidas de 50 neurônios cada, (50,100,50) com três camadas escondidas de 50, 100 e 50 neurônios respectivamente, e (100,) com uma camada escondida de 100 neurônios. O parâmetro *activation* especifica a função de ativação utilizada nas camadas escondidas, onde foram testadas tanto a função tangente hiperbólica (*tanh*) quanto a função *ReLU* (Rectified Linear Unit). O *solver*, que define o algoritmo utilizado para otimizar os pesos da rede neural, foi configurado para testar o gradiente estocástico descendente (*sgd*) e o otimizador *adam*. O parâmetro *alpha*, responsável pela regularização, foi testado com os valores 0.0001 e 0.05 para observar o efeito da regularização. Finalmente, o *learning_rate*, que determina a política de taxa de aprendizado, foi testado com valores constantes e adaptativos.

No modelo de Regressão Logística, utilizou-se o parâmetro C, que controla a força da regularização, testando os valores 0.1, 1.0 e 10.0. O parâmetro *penalty*, que especifica a norma usada na penalização, foi configurado para testar a penalização *L2*.

Para o modelo de SVM (*Support Vector Machine*), utilizou-se o parâmetro C para regularização, testando os valores 0.1, 1.0 e 10.0. O *kernel*, que define a função do kernel a ser utilizada, foi configurado para testar os kernels linear e radial basis function (*rbf*).

No modelo de Árvore de Decisão (*Decision Tree*), os parâmetros *max_depth* e *min_samples_split* foram ajustados. O parâmetro *max_depth*, que define a profundidade máxima da árvore, foi testado com os valores None, 10, 20 e 30. O parâmetro *min_samples_split*, que define o número mínimo de amostras exigidas para dividir um nó interno, foi testado com os valores 2, 10 e 20. Essas escolhas de parâmetros foram feitas para cobrir uma variedade de configurações possíveis e encontrar a melhor combinação que maximiza o desempenho dos modelos.

6.4.3 Resultados do Treinamento

A análise dos resultados do treinamento revelou que o modelo de Rede Neural Artificial apresentou o melhor desempenho geral, com uma AUC-ROC de 84% e superior aos demais modelos. A acurácia do modelo também foi elevada, indicando uma boa quantidade de verdadeiros positivos e negativos em relação à amostra.

O critério para julgar a qualidade dos algoritmos foi a AUC-ROC. A Figura 6.2 apresenta as curvas ROC para os diferentes modelos avaliados.

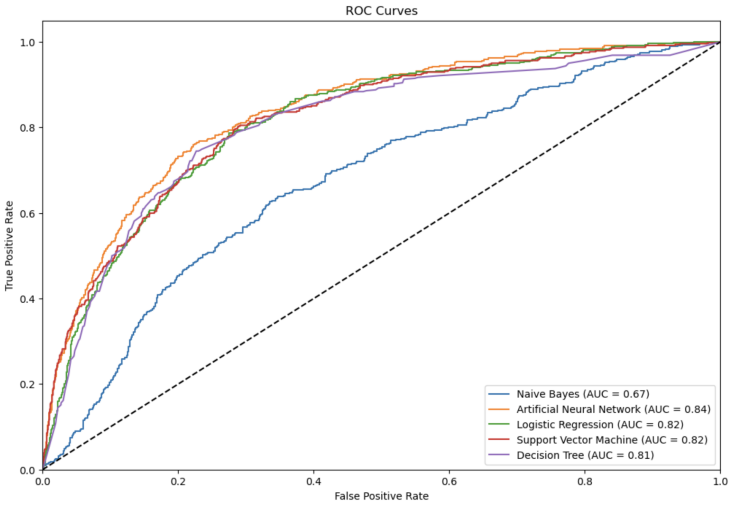

Figura 6.2: Curvas ROC dos Modelos Avaliados

A Tabela 6.1 resume os relatórios de classificação para cada modelo, destacando as métricas de acurácia, precisão, recall e f1-score. Esses relatórios fornecem uma visão detalhada sobre o desempenho de cada modelo, permitindo identificar pontos fortes e fracos em termos de previsões corretas e erros.

Modelo	**Acurácia**	**Precisão**	**Recall**	**F1-score**
Naïve Bayes	0.508	0.3021	0.7946	0.4416
Rede Neural Artificial	0.813	0.6406	0.5104	0.5678
Regressão Logística	0.7995	0.6294	0.4087	0.4982
Support Vector Machine	0.8005	0.6072	0.4876	0.5407
Árvore de Decisão	0.7965	0.5773	0.5809	0.5791

Tabela 6.1: Comparação dos Modelos de Machine Learning

Por fim, a matriz de confusão do melhor modelo, a Rede Neural Artificial, é apresentada na Tabela 6.2. A matriz de confusão ilustra a distribuição dos verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos,

fornecendo insights valiosos sobre a performance do modelo em termos de classificações corretas e erros.

	Previsto 0	Previsto 1
Verdadeiro 0	1380	138
Verdadeiro 1	236	246

Tabela 6.2: Matriz de Confusão - Rede Neural Artificial

A performance do modelo pode ser considerada relativamente satisfatória na identificação correta do comportamento dos usuários. É possível destacar que ele consegue identificar com eficiência usuários que não darão churn, o que é importante para a estratégia de comunicação em razão de dois motivos: o primeiro é que lançar comunicações para usuários que não estariam deixando de usar o produto é financeiramente custoso e pode, possivelmente, sobrecarregar e irritar usuários com a quantidade excessiva de mensagens; o segundo é que, caso haja uma estratégia de retenção que ofereça promoções e benefícios exclusivos para usuários prestes a darem churn, a identificação incorreta destes usuários pode representar uma perda financeira significativa para a empresa.

Embora a acurácia e a precisão sejam relativamente altas, indicando que a Rede Neural é eficaz na previsão correta de churn em muitos casos, o recall de 51,04% revela uma deficiência significativa, pois demonstra que o modelo está identificando apenas um pouco mais da metade dos usuários que realmente deram churn. Assim, é evidente que o modelo ainda possui espaço para melhorias, que serão exploradas no Capítulo 7 (Considerações Finais).

Dessa forma, pode-se concluir que, a partir da definição do critério de AUC-ROC como o determinante para a qualidade do modelo, a Rede Neural Artificial foi o algoritmo que apresentou os melhores resultados, contudo ainda há espaço para melhorias.

7
Considerações Finais

Este trabalho teve como objetivo desenvolver um modelo robusto de previsão de churn utilizando técnicas de Aprendizado de Máquina aplicadas no contexto empresarial, com foco na NG.CASH, uma carteira digital voltada para a Geração-Z. Através da implementação de uma infraestrutura escalável em nuvem, utilizando serviços da AWS como SageMaker, EMR Serverless, Step Functions e Lambda Functions, foi possível construir um pipeline completo para a coleta, processamento, treinamento e inferência de dados.

O desenvolvimento deste projeto proporcionou várias contribuições significativas. Primeiramente, foi fornecido um guia detalhado e prático sobre a implementação de modelos de Machine Learning em um ambiente empresarial, destacando a importância da computação em nuvem e boas práticas de Engenharia de Dados. Em segundo lugar, foi demonstrado na prática como as tecnologias de Machine Learning podem ser aplicadas para resolver problemas reais de negócios, especificamente a previsão de churn em uma instituição financeira digital. Além disso, a construção de uma infraestrutura escalável permite que o modelo possa ser facilmente adaptado e replicado para outros projetos e empresas, aumentando sua aplicabilidade e impacto.

Durante o desenvolvimento do projeto, diversos desafios foram enfrentados, como garantir a qualidade e disponibilidade dos dados, selecionar e ajustar os algoritmos de Machine Learning mais adequados e construir uma infraestrutura que pudesse lidar eficientemente com grandes volumes de dados. A automatização do Workflow também foi um desafio significativo, exigindo a integração coordenada de múltiplos serviços para criar pipelines complexas e eficientes.

Apesar de todo o trabalho contido no desenvolvimento desta pesquisa, é importante destacar que ainda há um espaço considerável para a implementação de melhorias. Primeiramente, aumentar a quantidade e melhorar a qualidade da seleção de atributos no script de pré-processamento pode impactar significativamente os resultados dos algoritmos de treino. Pode-se, por exemplo, incluir mais atributos como recorrência de transações *In* (aumento de saldo), gasto com serviços de assinatura como streaming e delivery, engajamento nas redes sociais, número de reclamações no suporte e muitos outros. Além disso, o envio de eventos para o ecossistema de dados também pode ser melhorado ao, por exemplo, aumentar o número de campos presentes no payload para melhorar a clusterização dos usuários nas plataformas de CRM e

personalizar ainda mais a comunicação. Finalmente, é imperativo implementar métodos para avaliar o impacto real das previsões de churn nas estratégias de retenção e na lucratividade da empresa, utilizando técnicas de experimentação A/B e criando dashboards de acompanhamento.

Em conclusão, este projeto demonstrou a viabilidade e a eficácia da utilização de técnicas avançadas de Aprendizado de Máquina e Computação em Nuvem para prever churn em um ambiente empresarial. A aplicação prática dessas tecnologias no contexto da NG.CASH forneceu insights valiosos e estratégias de retenção eficazes, ressaltando a importância de uma abordagem orientada a dados para o sucesso empresarial no cenário competitivo atual. Com as contribuições e aprendizados deste trabalho, espera-se que outras empresas possam adotar e adaptar essas práticas para melhorar suas operações e retenção de clientes, promovendo um uso mais inteligente e estratégico dos dados.

8
Referências bibliográficas

ABDOLRASOL, M. G. M. **Artificial Neural Networks Based Optimization Techniques: A Review — mdpi.com**. 2021. <https://www.mdpi.com/2079-9292/10/21/2689>. [Accessed 19-05-2024].

AFFAIRS, W. H. O. of C. **50 facts about customer experience for 2011**. 2011. Disponível em: <http://returnonbehavior.com/2010/10/50-facts-about-customer-experience-for-2011/>.

A.K., A. et al. Customer churn prediction in telecom using machine learning in big data platform. **J Big Data**, 2019.

BERRAR, D. **Bayes' Theorem and Naive Bayes Classifier**. 2001. <https://www.researchgate.net/profile/Daniel-Berrar/publication/324933572_Bayes'_Theorem_and_Naive_Bayes_Classifier/links/5d837aba92851ceb79143b04/Bayes-Theorem-and-Naive-Bayes-Classifier.pdf>. [Accessed 19-05-2024].

CERVANTES, J. et al. **A comprehensive survey on support vector machine classification: Applications, challenges and trends**. 2020. <https://scholar.google.com/scholar?hl=pt-BR&as_sdt=0%2C5&as_ylo=2020&q=support+vector+machines+for+classification&btnG=&oq=support+vector+machines>. [Accessed 20-05-2024].

CHKONIYA, V. Challenges in decoding consumer behavior with data science. **Department of Applied Mathematics, GOVCOPP, ISCA-UA, University of Aveiro, Portugal**, 2020.

DUTTA, P.; DUTTA, P.; XORIANT, P. M. I. Comparative study of cloud services offered by amazon, microsoft and google. **International Journal of Trend in Scientific Research and Development**, 2019. Disponível em: <https://api.semanticscholar.org/CorpusID:197881589>.

GARCIA, A. L. et al. A cloud-based framework for machine learning workloads and applications. **IEEE Access**, v. 8, p. 18681–18692, 2020.

GONZALEZ, L. de A. **Regressão Logística e suas Aplicações**. 2018. <https://monografias.ufma.br/jspui/bitstream/123456789/3572/1/LEANDRO-GONZALEZ.pdf>. [Accessed 20-05-2024].

GROGAN, J. et al. **A Multivocal Literature Review of Function-as-a-Service (FaaS) Infrastructures and Implications for Software Developers — link.springer.com**. 2020. <https://link.springer.com/chapter/10.1007/978-3-030-56441-4_5>. [Accessed 19-05-2024].

HANDELMA et al. **Peering Into the Black Box of Artificial Intelligence: Evaluation Metrics of Machine Learning Methods**. 2018. <https://sci-hub.se/10.2214/ajr.18.20224>. [Accessed 26-05-2024].

HARRIS, J. K. **Primer on binary logistic regression — ncbi.nlm.nih.gov**. 2021. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8710907/>. [Accessed 19-05-2024].

JOSHI, A. V. **Amazon's Machine Learning Toolkit: Sagemaker — link.springer.com**. 2019. <https://link.springer.com/chapter/10.1007/978-3-030-26622-6_24>. [Accessed 19-05-2024].

JOURNALS, E. **A REVIEW PAPER ON AWS by Abhishek Saini, Chaman Sharma , Nadeem Khan, Rohit Chauchan, Gurjeet Singh — eprajournals.com**. 2024. <https://eprajournals.com/IJMR/article/12110>. [Accessed 30-04-2024].

JUNIOR, E. E. R. **Estratégias para Classificação Binária Um estudo de caso com classificação de e-mails**. 2016. <https://jreduardo.github.io/ce064-ml/work-master.pdf>. [Accessed 26-05-2024].

KROGH, A. **What are artificial neural networks? - Nature Biotechnology — nature.com**. 2008. <https://www.nature.com/articles/nbt1386>. [Accessed 19-05-2024].

MARKETING, T. C. I. of. **Cost of Customer Acquisition versus Customer Retention**. 2010.

MARR, B. **Big Data in Practice: How 45 Successful Companies Used Big Data Analytics to Deliver Extraordinary Results**. 1. ed. Wiley, 2016. ISBN 1119231388,9781119231387. Disponível em: <http://gen.lib.rus.ec/book/index.php?md5=111cc53de5a24b3a4fd87e9994943727>.

MARTINS, B. d. A. M. B. S. **Regressão Logística ou Redes Neurais? Análise de Desempenho na Previsão de Atrito de Clientes em um Banco Comercial**. 2023. <http://www.repositorio.poli.ufrj.br/monografias/projpoli10040653.pdf>. [Accessed 20-05-2024].

MATHEW VASILIOS ANDRIKOPOULOS, F. J. B. A. **Exploring the cost and performance benefits of AWS step functions using a data processing pipeline | Proceedings of the 14th IEEE/ACM International Conference on Utility and Cloud Computing — dl.acm.org**. 2021. <https://dl.acm.org/doi/abs/10.1145/3468737.3494084>. [Accessed 19-05-2024].

MISHRA, A. **Machine Learning in the AWS Cloud : Add Intelligence to Applications with Amazon SageMaker and Amazon Rekognition**. John Wiley & Sons, Incorporated, 2019. ISBN 9781119556732,1119556732,9781119556718,9781119556725,1489729763,0059441380,0141228504,0867
Disponível em: <http://gen.lib.rus.ec/book/index.php?md5=58A7350438769CF46C09192B8E4ADA8F>.

MOSTAFA, A. **Customer Churn Prediction Model and Identifying Features to Increase Customer Retention based on User Generated Content**. 2020. Disponível em: <https://www.researchgate.net/profile/Alaa-Mostafa-15/publication/341872965_Customer_Churn_Prediction_Model_and_Identifying_Features_to_Increase_Customer_Retention_

based_on_User_Generated_Content/links/5f2c35eaa6fdcccc43b01626/
Customer-Churn-Prediction-Model-and-Identifying-Features-to-Increase-Customer-Retention-based-on-U
pdf>.

NASIR, S. Customer retention strategies and customer loyalty. 2018. Disponível em: <https://www.researchgate.net/publication/317001664_Customer_Retention_Strategies_and_Customer_Loyalty/links/5e43dfaaa6fdccd9659c14c2/Customer-Retention-Strategies-and-Customer-Loyalty.pdf>.

NASRI, A. L. X. G. **Development of Logistic Regression Models for Binary Classification of Covid-19 and Statistical Prediction of Deaths | Research, Society and Development — doi.org**. 2024. <https://doi.org/10.33448/rsd-v13i4.45446>. [Accessed 20-05-2024].

PANTIC, I. **Artificial neural networks in contemporary toxicology research — sciencedirect.com**. 2023. <https://www.sciencedirect.com/science/article/pii/S0009279722004744?casa_token=NmmSw_BI27EAAAAA:fyB2NLH9I8a_uymQJOOfqw4NF_43hRpT-i3YjXLoRi4bJewFsm-kr-mLzQrMvUM2MbuYYPVkyO4>. [Accessed 19-05-2024].

RAINIO JARMO TEUHO, R. K. O. **Evaluation metrics and statistical tests for machine learning - Scientific Reports — nature.com**. 2024. <https://www.nature.com/articles/s41598-024-56706-x>. [Accessed 26-05-2024].

REICHHELD, F. Prescription for cutting costs. **Bain & Company**, 2011.

RISH*, I. **An empirical study of the naive Bayes classifier**. 2019. <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=2825733f97124013e8841b3f8a0f5bd4ee4af88a>. [Accessed 19-05-2024].

SAEED SARAH BARAS, H. H. I. **Security and Privacy of AWS S3 and Azure Blob Storage Services**. 2019. <https://ieeexplore.ieee.org/abstract/document/8821735>. [Accessed 04-05-2024].

SAFAVIAN, S.; LANDGREBE, D. A survey of decision tree classifier methodology. **IEEE Transactions on Systems, Man, and Cybernetics**, v. 21, n. 3, p. 660–674, 1991.

SBARSKI, P. **Serverless Architectures on AWS: With examples using AWS Lambda**. 1. ed. Manning Publications, 2017. ISBN 1617293822, 978-1617293825. Disponível em: <http://gen.lib.rus.ec/book/index.php?md5=a188dbdbb7fdd81e6bc8ad30fd1757a9>.

SCHMIDT, C. P. K. **Programming Elastic MapReduce: Using AWS Services to Build an End-to-End Application**. O'Reilly Media, 2013. ISBN 978-1-44936-362-8. Disponível em: <http://gen.lib.rus.ec/book/index.php?md5=84ed27956d53d622dd7ceff0b8eb89bd>.

SHENG, J. et al. A binary classification study of alzheimer's disease based on a novel subclass weighted logistic regression method. **IEEE Access**, v. 10, p. 68846–68856, 2022.

SHEYKHMOUSA, M. **Support Vector Machine Versus Random Forest for Remote Sensing Image Classification: A Meta-Analysis and Systematic Review**. 2020. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9206124>. [Accessed 26-05-2024].

SINHA, P. **Cloud Computing Using AWS : An Analysis**. 2020. <http://gnanaganga.inflibnet.ac.in:8080/jspui/handle/123456789/1096>. [Accessed 27-04-2024].

V, U.; K, I. A survey on customer churn prediction in telecom industry: datasets, methods and metric. **Int Res J Eng Technol**, v. 3, n. 4, p. 1065–1070, 2016.