FACULDADE DE TECNOLOGIA DE PRESIDENTE PRUDENTE
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ÁNALISE DE ALGORITMOS DE APRENDIZADO DE MÁQUINA UTILIZANDO DADOS DO YOUTUBE

LEANDRO RONDONI DOS SANTOS

Presidente Prudente – SP
2024

FACULDADE DE TECNOLOGIA DE PRESIDENTE PRUDENTE
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ANÁLISE DE ALGORITMOS DE APRENDIZADO DE MÁQUINA UTILIZANDO DADOS DO YOUTUBE

LEANDRO RONDONI DOS SANTOS

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de Presidente Prudente, como requisito parcial para obtenção do diploma de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientadora: Profa. Mestra Vanessa dos Anjos Borges

Presidente Prudente – SP
2024

LEANDRO RONDONI DOS SANTOS

ANÁLISE DE ALGORITMOS DE APRENDIZADO DE MÁQUINA UTILIZANDO DADOS DO YOUTUBE

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de Presidente Prudente, como requisito parcial para obtenção do diploma de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Presidente Prudente, 22 de junho de 2024.

BANCA EXAMINADORA

Vanessa dos Anjos Borges
Orientador(a): Profa. Mestra Vanessa dos Anjos Borges
Faculdade de Tecnologia de Presidente Prudente
Presidente Prudente - SP

Adriane Cavichiolli
Profa. Mestra Adriane Cavichiolli
Faculdade de Tecnologia de Presidente Prudente
Presidente Prudente - SP

Sidinei de O. Sousa
Prof. Doutor Sidinei de Oliveira Sousa
Universidade do Oeste Paulista - UNOESTE
Presidente Prudente - SP

AGRADECIMENTO

À Profa. Mestra Vanessa dos Anjos Borges, meu sincero agradecimento por sua orientação competente e por acreditar no meu potencial. Seu apoio foi essencial para o desenvolvimento deste trabalho.

À minha mãe, Fatima, agradeço por todo o amor e apoio que sempre me ofereceu. Sua presença em minha vida é um pilar fundamental e suas palavras de encorajamento foram vitais para a minha trajetória acadêmica.

EPÍGRAFE

"A tecnologia é só uma ferramenta. No que se refere a motivar as crianças e conseguir que trabalhem juntas, um professor é um recurso mais importante."

(Bill Gates)

RESUMO

ANÁLISE DE ALGORITMOS DE APRENDIZADO DE MÁQUINA UTILIZANDO DADOS DO YOUTUBE

Este trabalho explora a aplicação das técnicas do aprendizado de máquina para analisar e interpretar dados coletados do *Youtube*. Com o constante crescimento do volume de conteúdo gerado por usuário no *Youtube*, surge a necessidade de classificar e prever de maneira eficiente a performance de um vídeo e o engajamento dos espectadores. O estudo explora vários algoritmos de classificação incluindo árvore de decisão, regressão linear e Máquina de Vetor Suporte (SVM), para categorizar os vídeos do *Youtube* baseado em vários atributos como as visualizações, *likes*, *dislikes* e a quantidade de comentários. Além da classificação, o artigo examina algoritmos de previsão voltado para prever a popularidade de um e o comportamento dos espectadores. A eficácia deste modelo é avaliada usando as métricas de Erro Médio Quadrático (MSE) e Coeficiente de Determinação. Estes métodos de avaliação fornecem uma compreensão abrangente da performance de cada modelo, destacando seus pontos fortes e suas limitações no tratamento dos dados dinâmicos e diversificados do *Youtube*. Os resultados ressaltam o potencial do aprendizado de máquina para melhorar a categorização e sistemas de recomendação no *Youtube*, oferecendo melhorias significantes na experiência do usuário e no gerenciamento de conteúdo. Esta pesquisa contribui para o campo mais amplo da ciência de dados, demonstrando aplicações práticas dos algoritmos de aprendizado de máquina em análise de mídias sociais, fornecendo insights valiosos para desenvolvedores, profissionais de marketing e criadores de conteúdo com o objetivo de alavancar estratégias orientadas por dados para otimizar sua presença no *Youtube*.

Palavras-chave: Ciência de dados. Machine Learning. Descoberta de Conhecimento.

ABSTRACT

ANALYSIS OF MACHINE LEARNING ALGORITHMS USING YOUTUBE API

This work explores the application of machine learning techniques to analyze and interpret data collected from the Youtube. With the constant growth in the volume of user-generated content on YouTube, there is a need to efficiently classify and predict a video's performance and viewer engagement. The study explores several classification algorithms including decision tree, linear regression and Support Vector Machine (SVM), to categorize YouTube videos based on various attributes such as views, likes, dislikes and the number of comments. In addition to rating, the article examines prediction algorithms aimed at predicting a viewer's popularity and behavior. The effectiveness of this model is evaluated using the Mean Squared Error (MSE) and Coefficient of Determination metrics. These evaluation methods provide a comprehensive understanding of each model's performance, highlighting its strengths and limitations in handling YouTube's dynamic and diverse data. The results highlight the potential of machine learning to improve categorization and recommendation systems on YouTube, offering significant improvements in user experience and content management. This research contributes to the broader field of data science by demonstrating practical applications of machine learning algorithms in social media analytics, providing valuable insights for developers, marketers, and content creators aiming to leverage data-driven strategies. to optimize your presence on Youtube.

Keywords: Data science. Machine learning. Knowledge Discovering.

SUMÁRIO

1 INTRODUÇÃO

O presente estudo aborda a aplicação de algoritmos de aprendizado de máquina utilizando dados obtidos do *YouTube*, uma das maiores plataformas de compartilhamento de vídeos do mundo. Com bilhões de usuários ativos, o *YouTube* oferece uma vasta quantidade de dados que pode ser analisada para entender melhor o comportamento dos usuários e as tendências emergentes.

Neste contexto, o uso de aprendizado de máquina se apresenta como uma ferramenta poderosa para processar e analisar grandes volumes de dados de maneira eficiente. Algoritmos avançados podem identificar padrões que seriam difíceis de detectar manualmente. Por meio dos dados fornecidos pelo *YouTube*, é possível coletar dados detalhados sobre os vídeos, como número de visualizações, curtidas, comentários etc., além de informações sobre os espectadores, como localização geográfica e preferências de conteúdo. Essa riqueza de dados permite a criação de modelos preditivos que podem ajudar os criadores de conteúdo a tomar decisões informadas sobre como melhorar a performance de seus vídeos.

A pesquisa tem como foco principal apresentar e demonstrar como o aprendizado de máquina pode auxiliar na compreensão e análise de grandes volumes de dados. Para alcançar esses objetivos, o estudo empregará algoritmos de aprendizado de máquina em várias etapas, incluindo coleta de dados, pré-processamento, seleção de variáveis, e avaliação de modelos. A aplicação de modelos preditivos e a avaliação de sua eficácia por meio de métricas, proporcionarão uma visão detalhada dos benefícios da utilização das técnicas de aprendizado de máquina na análise de grandes volumes de dados, destacando seu potencial em diversas áreas de estudo e aplicação prática.

2 JUSTIFICATIVA

Com o crescimento exponencial do conteúdo gerado por usuários no *Youtube* e bilhões de usuários ativos mensalmente e um volume massivo de novos vídeos todos os dias, surge assim o desafio e a necessidade de gerenciar e entender este enorme acervo de dados.

A aplicação de algoritmos de aprendizado de máquina oferece uma abordagem promissora para enfrentar este desafio. Estes algoritmos podem automatizar a classificação de vídeos com diversos atributos, facilitando a organização e a recomendação de conteúdo. Além disso, a previsão de métricas de desempenho, como visualizações, permite uma melhor compreensão das tendências de consumo de mídia e o planejamento de conteúdos futuros.

Juntamente com os algoritmos, é de extrema importância a utilização de métricas de avaliação de modelos, como a acurácia, para garantir e comprovar sua eficiência. Eles são como indicadores de qualidade do algoritmo e fornecem uma visão detalhada do desempenho dos algoritmos, destacando suas capacidades e limitações.

Este estudo é justificado pela necessidade de evidenciar como a utilização dos algoritmos de aprendizado de máquina nos auxilia na resolução de problemas, na otimização da gestão de conteúdo e no fornecimento de ideias tanto para criadores de conteúdo quanto para quem deseja se aprofundar no estudo de ciência de dados. Este trabalho contribui para a evolução das estratégias de análise de dados e reforça a importância do uso de tecnologias para resolver problemas contemporâneos em ambientes digitais dinâmicos.

3 OBJETIVOS

3.1 Objetivo Geral

O objetivo geral desta pesquisa é explorar e demonstrar a eficácia dos algoritmos de aprendizado de máquina na análise de grandes volumes de dados fornecidos pelo *Youtube*, visando otimizar a visibilidade e relevância na criação de conteúdo na plataforma.

3.2 Objetivos Específicos

1. Coletar dados: Obter dados detalhados sobre vídeos do *Youtube*, incluindo métricas de visualizações, *likes*, *dislikes*, comentários, através dos dados fornecidos pelo *Youtube*.
2. Pré-processar os dados: Realizar a limpeza, formatação, organização e transformação dos dados coletados, tratando valores ausentes e preparando os dados para análise.
3. Realizar análise exploratória dos dados: Realizar uma análise exploratória com a utilização de diferentes tipos de gráficos para obter uma compreensão inicial das principais características e padrões presentes.
4. Analisar e identificar padrões e tendências: Utilizar algoritmos de aprendizado de máquina para identificar padrões e tendências nos dados.
5. Avaliar modelos preditivos: Aplicar e avaliar modelos preditivos utilizando métricas de desempenho, para mediar a eficácia dos algoritmos na previsão dos dados.
6. Registar e interpretar os resultados: Registrar os resultados obtidos e interpretá-los, fornecendo uma análise sobre como os dados podem ser utilizados para otimizar a produção e promoção de conteúdo no *Youtube*.

4 REVISÃO DE LITERATURA

4.1 Aprendizado de máquina

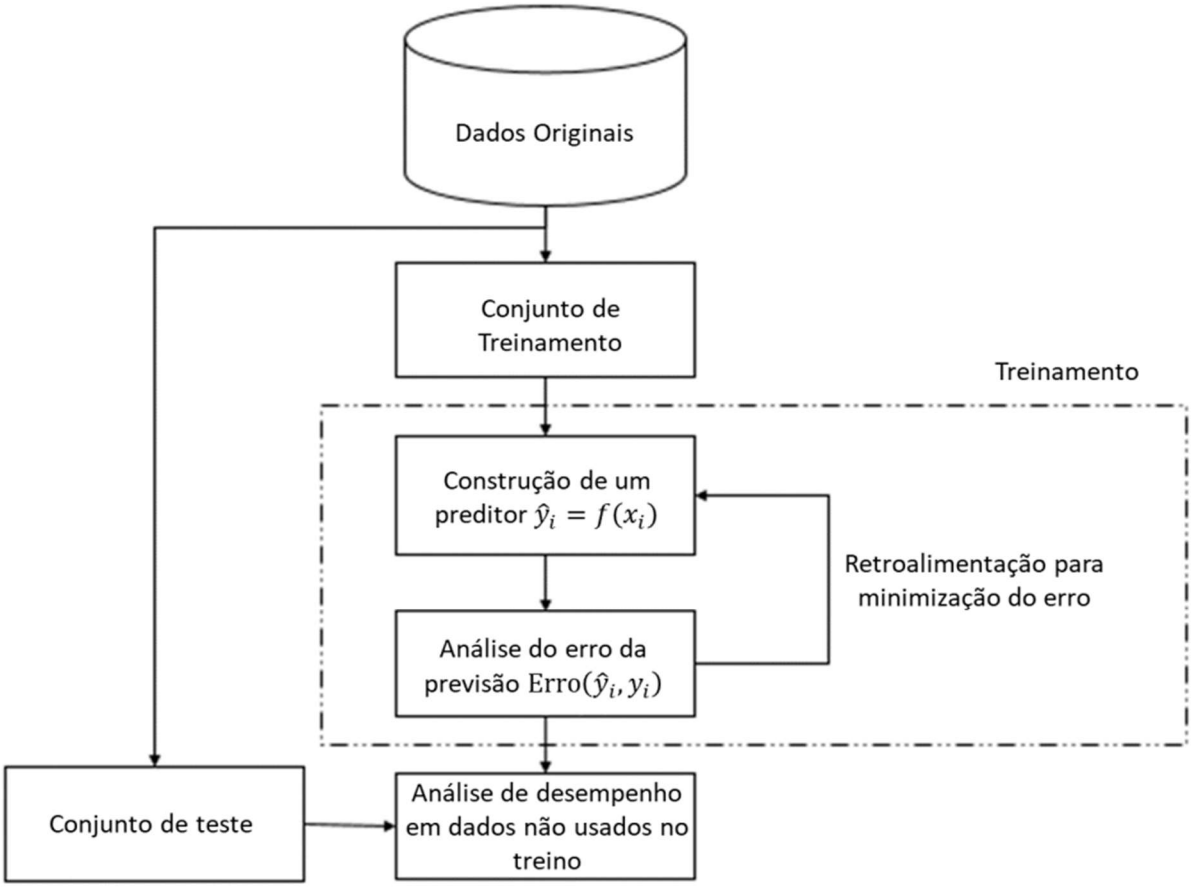
O aprendizado de máquina (AM) é uma disciplina da ciência da computação que se concentra no desenvolvimento de algoritmos e modelos que permitem que os sistemas aprendam e melhorem com a experiência. Um dos conceitos fundamentais do AM é a noção de que os computadores podem aprender sem serem explicitamente programados. Esse aprendizado é baseado em dados, onde os modelos são treinados em conjuntos de dados para fazer previsões ou tomar decisões.

Segundo Jordan e Mitchel (2015), um problema de aprendizado pode ser definido como o problema de otimizar alguma medida de performance ao executar alguma tarefa, através de algum tipo de experiência de treinamento. Além disso, o AM lida com a ideia de generalização, ou seja, a capacidade de um modelo de aplicar o conhecimento aprendido a novos dados não vistos anteriormente.

Existem vários tipos de aprendizado de máquina, incluindo o aprendizado supervisionado, onde os modelos são treinados em pares de entrada e saída, o aprendizado não supervisionado, que envolve a extração de padrões e estruturas dos dados sem rótulos explícitos, e o aprendizado por reforço, em que os modelos aprendem a tomar ações para maximizar uma recompensa em um ambiente específico (Holmes; Bialik; Fadel, 2019).

O aprendizado supervisionado é um dos métodos mais utilizados. Alguns sistemas que exemplifica muito bem este tipo de AM são: classificadores de *spam* dos e-mails, reconhecimento facial de imagens e sistemas de diagnósticos médicos para pacientes. Todos eles são treinados com uma coleção de pares de dados *(x,y)* e o objetivo produz uma predição *y* como uma resposta (Sathya; Abraham, 2013). A Figura 1 representa os passos básicos para o desenvolvimento de algoritmos de aprendizado supervisionado.

Figura 1: Funcionamento de Algoritmos de Aprendizado Supervisionado



Fonte: Adaptado de Nguyen et al. (2016)

O aprendizado não supervisionado trabalha com dados não rotulados, analisando as propriedades de sua estrutura, tentando encontrar padrões ocultados ou grupos de dados. Este método é ideal para análise exploratória de dados, reconhecimento de padrões, estratégias de vendas cruzadas e também para reduzir o número de recursos em um modelo (Sathya; Abraham, 2013).

Jordan e Mitchel (2015) destacam que com o uso da Inteligência Artificial (IA), o AM emergiu como melhor método para o desenvolvimento de softwares para visão computacional, processamento de linguagem natural e jogos. Além disso, o aprendizado profundo, ou redes neurais profundas, tem se destacado nos últimos anos.

O aprendizado por reforço, por sua vez, não é treinado com dados de amostra. Este método aprende conforme seu avanço por meio de tentativa e erro, ou seja, quando o algoritmo encontrar uma sequência de respostas satisfatórias ele irá reforçá-

las para desenvolver uma melhor maneira ou recomendar uma melhor solução para o problema (Russel; Norvig, 2013).

As aplicações do aprendizado de máquina são amplas e abrangem diversos setores, como saúde, finanças, transporte, manufatura e muitos outros. Por exemplo, em medicina, o AM pode ser usado para auxiliar no diagnóstico de doenças com base em imagens médicas. No entanto, o AM também enfrenta desafios, como a interpretação dos modelos, questões éticas relacionadas ao uso de dados sensíveis, e a necessidade de lidar com o viés presente nos dados de treinamento (Freitas; Santana, 2019).

4.2 Algoritmos de Aprendizado de Máquina

4.2.1 ÁRVORES DE DECISÃO

Árvores de decisão são modelos que dividem o conjunto de dados em subconjuntos com base nas características, criando uma estrutura de árvore para tomar decisões de classificação.

Segundo Mitchell (1997, p. 52)

> árvores de decisão classificam instâncias ordenando–as da raiz para algum nó de folha, que fornece a classificação da instância. Cada nó da árvore especifica um teste de algum atributo da instância e cada ramo descendente deste nó corresponde a um dos possíveis valores para este atributo. Uma instância é classificada iniciando do nó raiz da árvore, testando o atributo especificado pelo nó e, em seguida, movendo-se para baixo do ramo de árvore correspondente ao valor do atributo no exemplo fornecido.

4.2.2 NAIVE BAYES

O algoritmo *Naive Bayes* é baseado no Teorema de *Bayes* que, segundo Mitchell (2013), “fornece uma maneira de calcular a probabilidade de uma hipótese baseada em sua probabilidade prévia, as probabilidades de observar vários dados dada uma hipótese e o próprio dado observado.”

Este tipo de algoritmo é frequentemente usado em tarefas de classificação de texto, como filtragem de spam. Ele assume independência condicional entre as

características, o que pode ser uma simplificação excessiva, mas muitas vezes funciona bem na prática devido à sua eficiência computacional e desempenho satisfatório em muitos cenários.

4.2.3 SUPPORT VECTOR MACHINES (SVM)

As *Support Vector Machines* (SVMs) são algoritmos que buscam encontrar um hiperplano (linha) de separação ótimo entre duas classes, maximizando a margem entre os pontos de dados mais próximos das fronteiras de decisão. SVMs são eficazes em espaços de alta dimensionalidade e podem lidar com conjuntos de dados não linearmente separáveis.

Gareth et al. (2023) ressalta que as SVMs são uma generalização de um simples e intuitivo classificador chamado classificador de margem máxima, mas infelizmente não pode ser aplicado na maioria dos conjuntos de dados, uma vez que, as classes precisam ser separáveis por uma fronteira linear.

4.2.4 K-NEAREST NEIGHBORS (K-NN)

O algoritmo k-*Nearest Neighbors* (k-NN) é um método simples, mas poderoso, que classifica novos pontos de dados com base na maioria das classes dos k vizinhos mais próximos no espaço de características. Ele é especialmente eficaz em problemas onde a estrutura dos dados é importante, como reconhecimento de padrões, classificação de documentos e previsão de séries temporais.

Segundo Gareth et al (2023), para dados reais, nós não sabemos a distribuição condicional de Y dado X, e então computando o classificador *Bayes* é impossível... Dado um inteiro positivo K e uma observação teste x0, o classificador KNN primeiro identifica os pontos K nos dados de treinamento que estão próximos de x0.

4.2.5 RANDOM FOREST

O algoritmo Random Forest é amplamente reconhecido por sua capacidade de melhorar a precisão das previsões em problemas de classificação e regressão. Baseado em uma abordagem de conjunto, o Random Forest combina várias árvores de decisão para criar um modelo robusto e preciso.

Como destacado por Liaw et al. (2002), este método incorpora a aleatoriedade na construção das árvores, selecionando aleatoriamente subconjuntos de dados de treinamento e características em cada etapa, mitigando assim o problema de *overfitting* comum em árvores de decisão individuais. Além disso, a combinação das previsões de múltiplas árvores de decisão por meio de votação majoritária (no caso de classificação) ou média (no caso de regressão) resulta em um modelo mais estável e geralmente mais preciso.

O *Random Forest* tem se mostrado eficaz em uma variedade de aplicações, incluindo diagnóstico médico, detecção de fraudes e previsão do mercado de ações, tornando-se uma escolha popular na comunidade de aprendizado de máquina.

4.2.6 REGRESSÃO LINEAR

A regressão linear, segundo Gareth et.al (2023), é um dos pilares do aprendizado de máquina e da análise estatística. Este algoritmo é amplamente utilizado para modelar as relações entre variáveis independentes e dependentes por meio de uma linha reta, permitindo previsões de valores contínuos. A principal referência para a regressão linear é a equação da reta, onde coeficientes são ajustados para minimizar a soma dos quadrados das diferenças entre os valores previstos e observados. Os coeficientes estimados representam a magnitude e a direção da influência das variáveis independentes nas variáveis dependentes.

Além disso, é uma ferramenta fundamental para a análise exploratória de dados e o estabelecimento de previsões precisas em muitos cenários do mundo real. Portanto, a regressão linear desempenha um papel importante na modelagem estatística e é uma base essencial para quem busca compreender e aplicar técnicas de aprendizado de máquina.

4.2.7 REGRESSÃO LOGÍSTICA

Conforme explicado por Gareth et.al(2023), a regressão linear é um algoritmo de aprendizado de máquina fundamental amplamente utilizado em problemas de classificação. Ele é particularmente eficaz quando o objetivo é modelar a probabilidade de um evento binário ocorrer, como a classificação de e-mails como spam ou não spam. A regressão logística utiliza a função logística para mapear

valores contínuos para o intervalo entre 0 e 1, representando a probabilidade da classe positiva. Esse método é altamente interpretável, permitindo a análise dos efeitos das variáveis independentes sobre a probabilidade do evento, tornando-se uma ferramenta valiosa em muitos domínios, desde medicina até finanças.

A abordagem da regressão logística é especialmente adequada para tarefas em que se deseja entender e comunicar as influências das variáveis independentes de forma clara e precisa. Seus coeficientes estimados fornecem informações sobre a direção e a magnitude dessas influências, tornando-a uma escolha frequente para problemas de classificação e análise de risco, onde a interpretabilidade é crucial, tornando-se uma ferramenta poderosa para prever eventos binários e extrair insights valiosos a partir de dados.

4.2.8 REDES NEURAIS

Redes neurais é uma classe avançada de algoritmos que buscam reproduzir o funcionamento do cérebro humano. São compostas por camadas de neurônios artificiais interconectados, que processam informações em paralelo para realizar tarefas complexas de reconhecimento de imagem, processamento de linguagem natural e análise de séries temporais.

Gareth et al.(2023) destaca que as redes neurais são altamente adaptáveis e podem ser empregadas em diversas áreas pelo seu grande poder de generalização, permitindo fazer previsões precisas em dados não vistos. Os dados de treinamento de redes neurais é uma das etapas mais desafiadoras de redes neurais, pois requer uma gama muito grande de dados para conseguir representar todas o máximo de variações possíveis de determinado dado.

4.3 Avaliação de modelos

A avaliação de modelos desempenha um papel crítico no campo do aprendizado de máquina e é essencial para medir o desempenho e a eficácia dos algoritmos de classificação e previsão. Várias métricas são amplamente utilizadas para avaliar modelos, incluindo a matriz de confusão, acurácia, precisão, recall e F1-score. Para avaliação de modelos de Regressão, os mais utilizados são o Erro Quadrático Médio (MSE) e o Coeficiente de Determinação (R^2).

4.3.1 ERRO QUADRÁTICO MÉDIO (MSE)

O Erro Quadrático Médio (MSE) é uma medida utilizada para avaliar a proximidade de um estimador em relação ao valor verdadeiro de um parâmetro. Ele é definido como a expectativa do quadrado do desvio do estimador em relação ao parâmetro verdadeiro. Em termos mais simples, MSE reflete tanto o viés (precisão) do estimador, ou seja, o quanto seu valor esperado desvia sistematicamente do valor verdadeiro, quanto a variância (exatidão) do estimador, que indica quanto ele varia em torno do valor esperado devido à variabilidade da amostra (Schluchter, 2005).

A Figura 2 apresenta a fórmula utilizada para desenvolver esse cálculo.

Figura 2: Fórmula para cálculo do Erro Quadrático Médio

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Fonte: Schluchter (2005)

4.3.2 COEFICIENTE DE DETERMINAÇÃO

O Coeficiente de Determinação, também conhecido como R^2, é uma medida de ajuste para modelos que se baseia na proporção de variância explicada. Ele é definido como o quadrado do coeficiente de correlação múltipla entre as variáveis dependentes e independentes. O R^2 mede a proporção da variação no valor dependente que é explicada pelas variáveis independentes no modelo. Esse coeficiente é amplamente utilizado em regressões lineares para avaliar a qualidade do ajuste do modelo, mas sua interpretação deve ser feita com cautela, especialmente em modelos que envolvem multicolinearidade ou outras complicações (Barret, 1974).

A Figura 3 apresenta a fórmula utilizada para desenvolver esse cálculo.

Figura 3: Fórmula para cálculo do Coeficiente de Determinação

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Fonte: Barret (1974)

4.4 Pontos de Atenção em Ciência de Dados

4.4.1 *Outliers*

Outliers podem ser definidos como observações que estão muito distantes dos demais valores de uma amostra, que parecem não fazerem parte do padrão de variabilidade produzido em outras variações (Draper; Smith, 1998; Johnson et al., 2002).

Para Cumbi (2010, p. 1)

> A presença de outliers nos dados de uma amostra provoca uma distorção nas estimativas, como é o caso da média aritmética, do desvio padrão (visto que estas sofrem maior influência dos valores extremos), nos modelos de regressão (que afecta a estimativa do coeficiente de correlação), o que pode provocar a não fiabilidade dos resultados nas inferências.

4.4.2 *Underfitting* e *Overfitting*

Um modelo de Aprendizagem de Máquina, ao serem treinados e testados, deve possuir os mesmos erros de treinamento e de teste.

A capacidade de um modelo pode ser muito alta, o que significa que o algoritmo escolhe uma função complexa que se encaixa perfeitamente nos dados de treinamento, mas falha em generalizar, porque superestima a importância de dados ruidosos. Esse comportamento é agrupado *overffiting* (Kollmannsberger, 2021).

Já, quando um modelo com uma capacidade muito baixa é aplicado a um problema altamente não-linear, por exemplo, se o espaço de hipótese de um modelo contiver apenas funções lineares, ele não poderá representar dados que estejam

seguindo funções quadráticas. Este caso descreve o *underfitting* (Kollmannsberger, 2021).

5 METODOLOGIA

O presente trabalho se propôs a analisar dados fornecidos pelo *Youtube*. Para alcançar o objetivo mencionado, foram empregados algoritmos de aprendizado de máquina utilizando a Linguagem de Programação Python. Desta maneira, os dados fornecidos pelo *Youtube* passaram por uma série de etapas conforme descrito a seguir.

A primeira etapa consistiu na coleta de dados e no conhecimento de dados. A coleta foi efetuada a partir de um repositório de dados no Kaggle (https://www.kaggle.com/datasets/datasnaek/youtube-new). O Kaggle.com é uma plataforma online muito reconhecida e utilizada por cientistas de dados e entusiastas da análise de dados, oferecendo um ambiente colaborativo para o desenvolvimento de projetos de *machine learning* e ciência de dados. A plataforma permite que os usuários compartilhem seus códigos, discutam abordagens e solucionem problemas, além de oferecer um espaço para construção de portfólios profissionais e a interação com uma comunidade global da área de ciência de dados.

O *Youtube* disponibiliza dados sobre seus vídeos categorizados por países. Foram, portanto, analisados cerca de 120.000 vídeos em 8 países diferentes (Canadá, Dinamarca, França, Grã-Bretanha, Índia, México, Rússia e Estados Unidos), abrangendo um período 8 meses, entre 14 de novembro de 2017 a 14 de junho de 2018. Nesse sentido, este trabalho aborda o estudo de uma série de dados temporais, definida como uma sequência de pontos de dados coletados ou registrados ao longo do tempo em intervalos regulares.

Conhecidos os dados, a preocupação se torna em preparar estes dados para serem analisados. Conhecida como pré-processamento, esta etapa é considerada por muitos, a etapa mais importante de todo processo. Os dados precisam ser adequados e formatados, os dados faltantes precisam receber um tratamento, assim como os dados que não se comportam como a grande maioria, os *outliers*. Depois deve selecionar e excluir a variáveis que ajudarão a responder nossa pergunta objetivo e por fim reduzir a dimensão dos dados.

Feito isto, é preciso escolher um modelo de aprendizado de máquina e aplicá-lo aos dados. Esta etapa se resume em escolher o algoritmo de aprendizado de máquina que será utilizado e executá-lo utilizando os dados a serem analisados pelo modelo escolhido.

Após a aplicação do modelo, a eficácia dos resultados obtidos foi avaliada por meio de métricas predefinidas. Por se tratar de dados caracterizados como séries temporais, as métricas utilizadas para a avaliação dos modelos incluem e o Erro Médio Quadrático e o Coeficiente de Determinação, que fornecem uma compreensão abrangente da capacidade do modelo se ajustar aos dados e atingir os objetivos estabelecidos anteriormente.

Os algoritmos escolhidos para alcançar o objetivo proposto foram as Árvores de Decisão, K-Nearest Neighbors, Regressão Linear e SVM.

Os modelos relatados na Tabela 1 foram utilizados a partir de suas implementações na linguagem de programação Python:

Tabela 1: Análise dos algoritmos utilizados

Algoritmo	Descrição	Vantagem	Desvantagem
Decision Tree	Fazem a captação da relação não linear entre variáveis identificam características importantes entre dados	Tem capacidade de lidar com série de dados categóricos e temporais	É propenso a overfitting em conjunto de dados pequenos.
Gradiente Boosting	Combina a simplicidade das árvores de decisão com a capacidade de formar modelos preditivos	Alta precisão e facilidade com lidar com dados heterogêneos	É computacionalmente caro
K-Nearest Neighbors (KNN)	É um modelo baseado em instâncias, que pode capturar relações complexas nos dados	Não requer treinamento intensivo	Ineficiente em termos de tempo de execução de grandes conjuntos de dados
Linear Regression	Abordagem mais simples para modelar relação entre variáveis dependentes e independentes	Simplicidade e facilidade de interpretação	Incapaz de modelar relações não lineares
Random Forest	Combina múltiplas árvores de decisão para previsão	Alta precisão e robustez contra *overfitting*	Pode ser mais difícil de interpretação em relação a árvores únicas
Support Vector Regression (SVR)	Extensão do SVM para problemas de regressão	Alta precisão e eficácia para dados não lineares	É computacionalmente caro

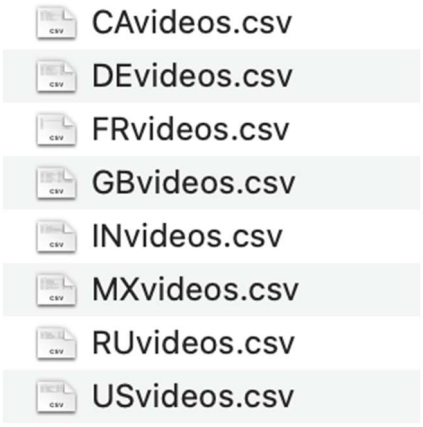
Fonte: Elaborado pelo autor

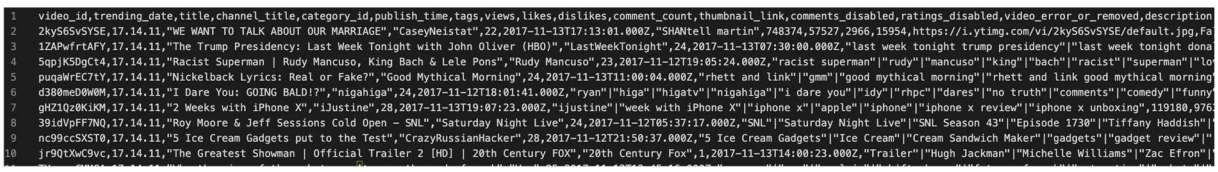
6 RESULTADOS E DISCUSSÃO

Após a análise dos dados coletados fornecidos pelo *Youtube*, os alvos foram definidos a partir das estatísticas dos vídeos como visualizações, *likes*, *dislikes* e quantidade de comentários.

Na Figura 4 é demonstrado todos os arquivos utilizados para a importação dos dados e a estrutura dos dados apresentados. Os arquivos contêm 17 colunas cada, com as seguintes informações: id do vídeo, data de tendência, título do vídeo, título do canal, id da categoria, data da publicação, *tags*, visualizações, *likes*, *dislikes*, quantidade de comentários, *link* da *thumbnail*, comentários desabilitados, avaliações desabilitadas, erro de vídeo ou removido, descrição e país.

Figura 4: Arquivos utilizados para análise dos dados e exemplo do conteúdo disponibilizado e cada um deles





Fonte: Elaborado pelo Autor

Foi utilizado a linguagem de programação Python para realização de todas as tarefas realizadas, desde a leitura dos dados até os cálculos utilizando vários modelos de aprendizado de máquina.

Após a importação dos dados, foi realizado a leitura e organização dos dados em tabelas, sendo uma tabela para cada país. Os países analisados, bem como suas respectivas siglas são listados abaixo.

1. CA: Canadá;
2. DE: Dinamarca;
3. FR: França;
4. GB: Grã-Bretanha;
5. IN: Índia;
6. MX: México;
7. RU: Rússia;
8. Estados Unidos.

A Figura 5 apresenta o código utilizado para importar as bibliotecas para o desenvolvimento do trabalho, além do código-fonte utilizado para importação dos arquivos.

Figura 5: Bibliotecas para análise dos dados e código-fonte para importação de dados

```python
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import mean_squared_error
import seaborn as sns
import matplotlib.pyplot as plt

pasta = './data'
files = [f for f in os.listdir(pasta) if f.endswith('.csv')]

dataframes = {}

def read_file(file_path):
    encondings = ['utf-8', 'latin1', 'iso-8859-1']
    for enc in encondings:
        try:
            return pd.read_csv(file_path, encoding=enc)
        except UnicodeDecodeError:
            continue


for i, arq in enumerate(files):
    file_path = os.path.join(pasta, arq)
    df = read_file(file_path).drop_duplicates(subset=['video_id'])
    df['country'] = arq[:2]
    if df is not None:
        var_name = f'df_{arq[:2]}'
        dataframes[var_name] = df

tabelas = {}
ys = {}

for v_name, data in dataframes.items():

    # Separando valores e Filtrando colunas
    X = data[['views', 'likes', 'dislikes', 'comment_count']].copy()
    titles = data[['video_id', 'trending_date', 'publish_time','title', 'category_id', 'country']].copy()

    # Juntado os dataframes
    X = titles.join(X)

    var_name = f'X_{v_name}'
    ys[var_name] = data['views']
    tabelas[var_name] = X
```

Fonte: Elaborado pelo autor

Para uma melhorar compreensão das características originais dos dados e planejamento de futuras transformações foi realizada uma análise exploratória utilizando gráficos dos tipos histograma, dispersão (*scatterplot*) e mapas de calor (*heatmaps*).

Histogramas e *scatterplot* foram utilizados para analisar a distribuição da frequência das caraterísticas analisadas. Os *heatmaps* foram utilizados para compreender como as variáveis analisadas se relacionam entre si. E, finalmente, os *boxplots* auxiliam na visualização da distribuição dos dados e na identificação dos outliers.

Quando um histograma apresenta os dados em formato de uma curva gaussiana, isso pode significar que os dados possuem uma distribuição normal.

Uma curva Gaussiana ou curva de sino, é uma função matemática que descreve a distribuição de uma variável contínua.

As Figuras 6, 7, 8 e 9 apresentam o código fonte e os histogramas dos dados dos diferentes países analisados.

Na Figura 6 é demonstrado a quantidade comentários nos 8 países analisados. Pode-se notar que os dados estão como uma distribuição normal pela formação de uma curva Gaussiana. Nos casos de França e México, é possível notar uma leve assimetria à esquerda.

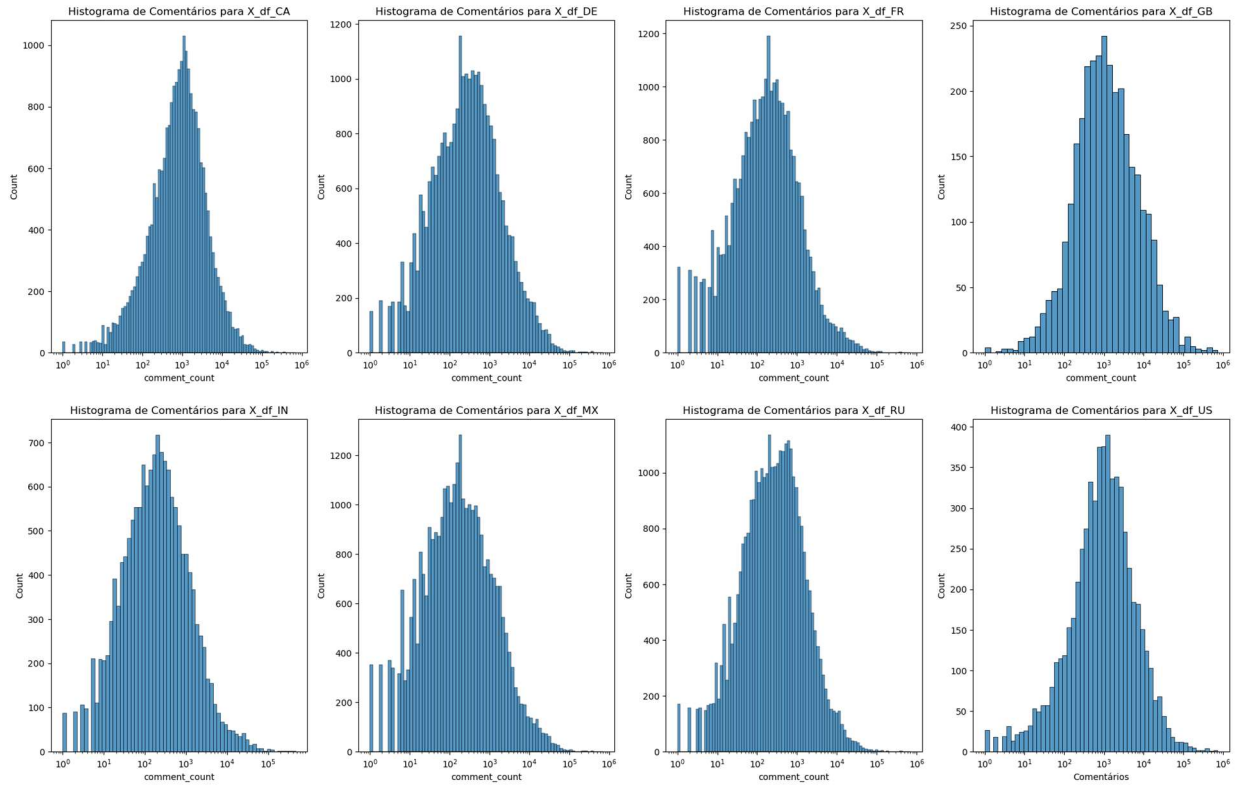
Figura 6: Código-fonte e Histograma dos dados de comentários

```python
#histograma comentários
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(24, 15))
# columns_to_plot = ['views', 'likes', 'dislikes', 'comment_count']

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    clean_df = df[np.isfinite(df['comment_count']) & (df['comment_count'] > 0)]

    sns.histplot(data=clean_df, x='comment_count', ax=ax, log_scale=True)
    ax.set_title(f'Histograma de Comentários para {nome}')
    plt.xlabel('Comentários')
    plt.ylabel('Count')

plt.show()
```



Fonte: Elaborado pelo autor

Na Figura 7 são demonstrados os histogramas de *dislikes*. A curva Gaussiana é mais definida no Canadá, na Grã-Bretanha, na Índia e nos Estados Unidos.

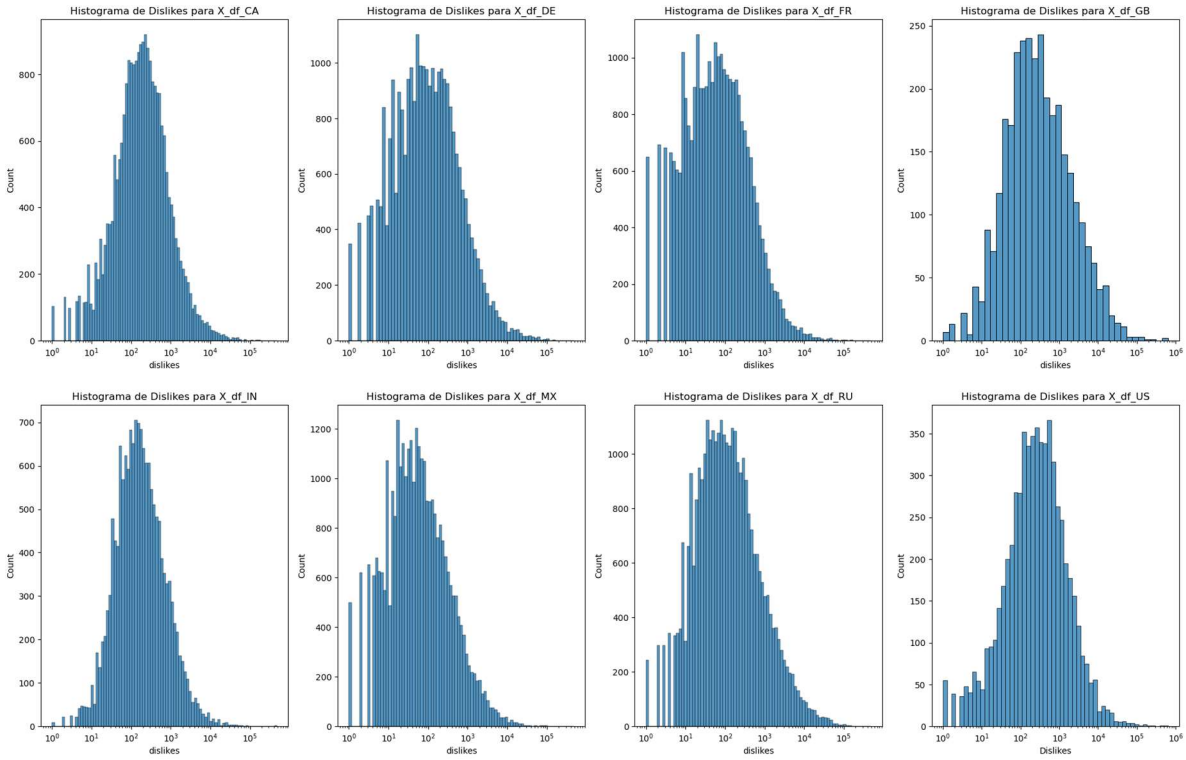
Figura 7: Código-fonte e Histograma dos dados de *dislikes*

```
#histograma dislikes
dor Remoto
; = plt.subplots(nrows=2, ncols=4, figsize=(24, 15))
# columns_to_plot = ['views', 'likes', 'dislikes', 'comment_count']

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    clean_df = df[np.isfinite(df['dislikes']) & (df['dislikes'] > 0)]

    sns.histplot(data=clean_df, x='dislikes', ax=ax, log_scale=True)
    ax.set_title(f'Histograma de Dislikes para {nome}')
    plt.xlabel('Dislikes')
    plt.ylabel('Count')

plt.show()
```



Fonte: Elaborado pelo autor

Na Figura 8 e 9 são demonstrados os histogramas de *likes* e visualizações (*views*) e pode-se notar claramente a curva Gaussiana em todos os países analisados.

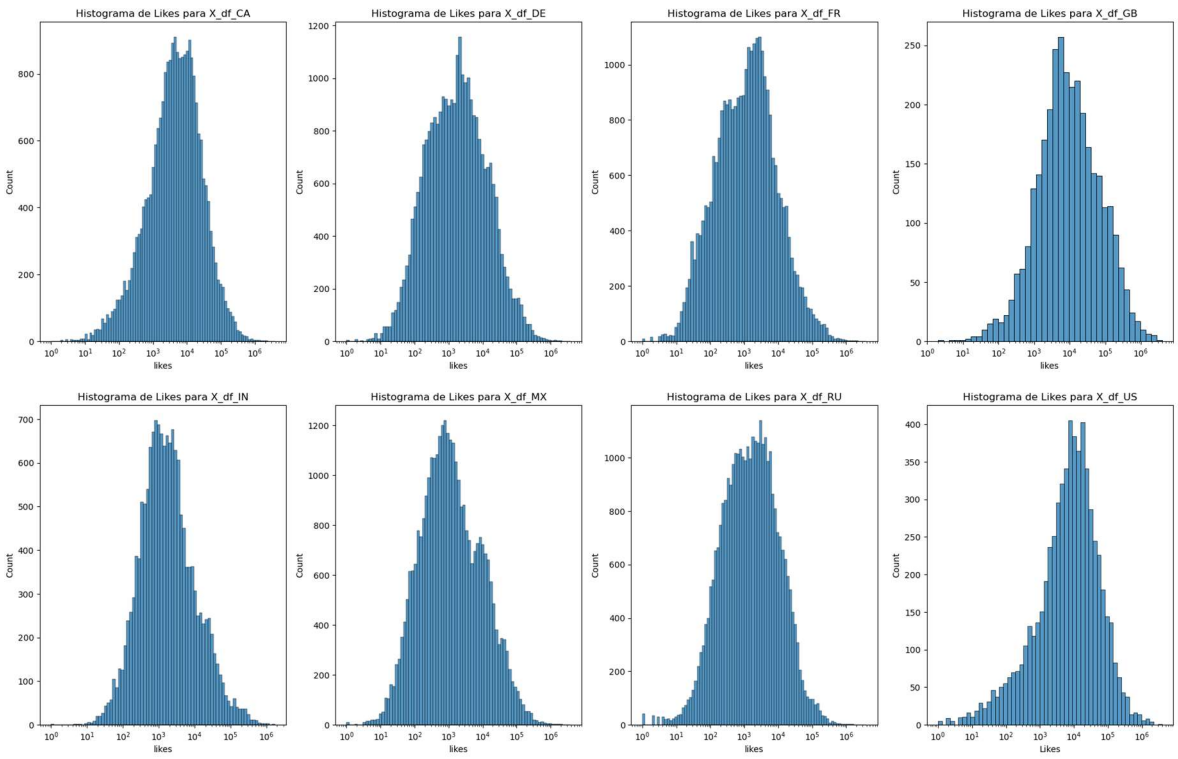
Figura 8: Código-fonte e Histograma dos dados de *likes*

```
#histograma likes
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(24, 15))

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    clean_df = df[np.isfinite(df['likes']) & (df['likes'] > 0)]

    sns.histplot(data=clean_df, x='likes', ax=ax, log_scale=True)
    ax.set_title(f'Histograma de Likes para {nome}')
    plt.xlabel('Likes')
    plt.ylabel('Count')

plt.show()
```



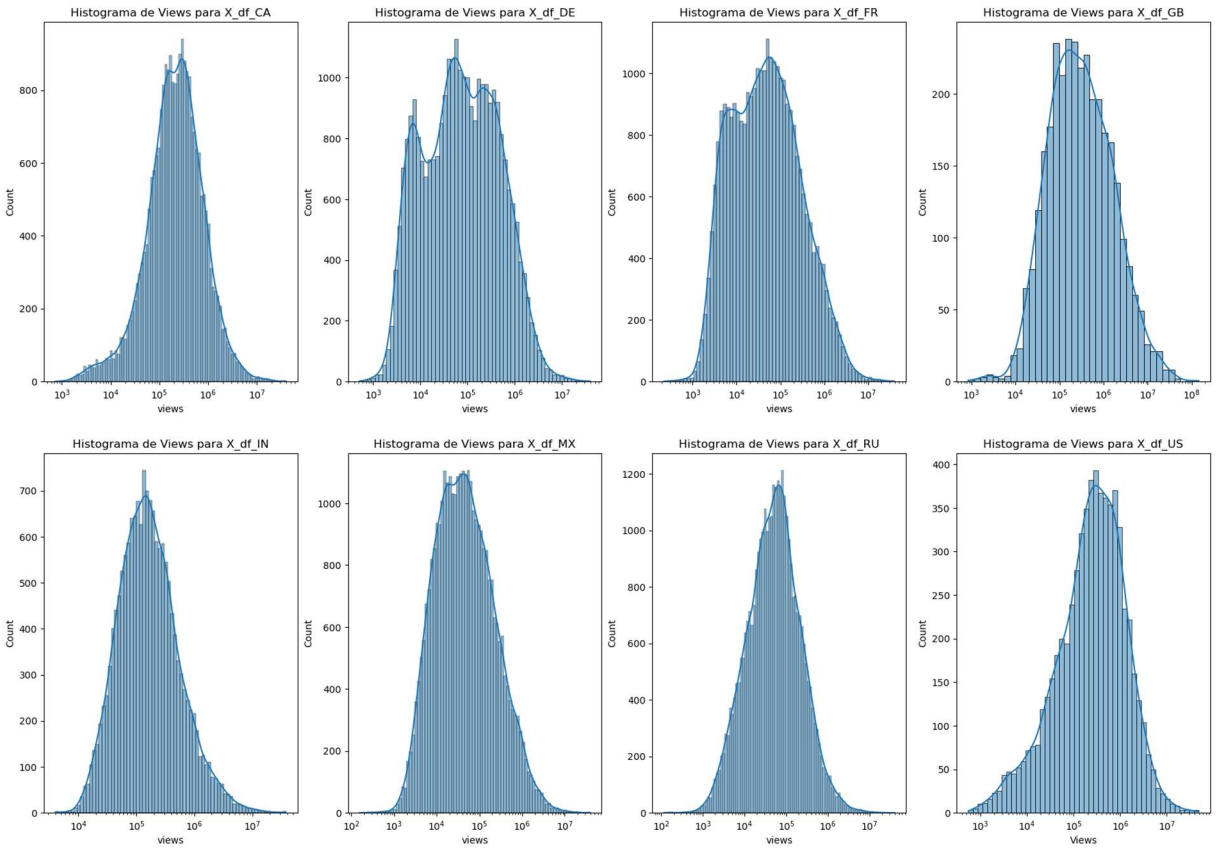
Fonte: Elaborado pelo autor

Figura 9: Código-fonte e Histograma dos dados de visualizações

```
#histograma views
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(22, 15))
columns_to_plot = ['views', 'likes', 'dislikes', 'comment_count']

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    sns.histplot(data=df, x='views', log_scale=True, ax=ax, kde=True)
    ax.set_title(f'Histograma de Views para {nome}')
    plt.xlabel('Views')
    plt.ylabel('Count')

plt.show()
```



Fonte: Elaborado pelo autor

A partir desses dados é interessante verificar que os dados estão em sua maioria normalmente distribuídos.

Entre as Figuras 10 a 14, são demonstrados o código-fonte e vários diagramas de dispersão representando a relação entre as diferentes variáveis.

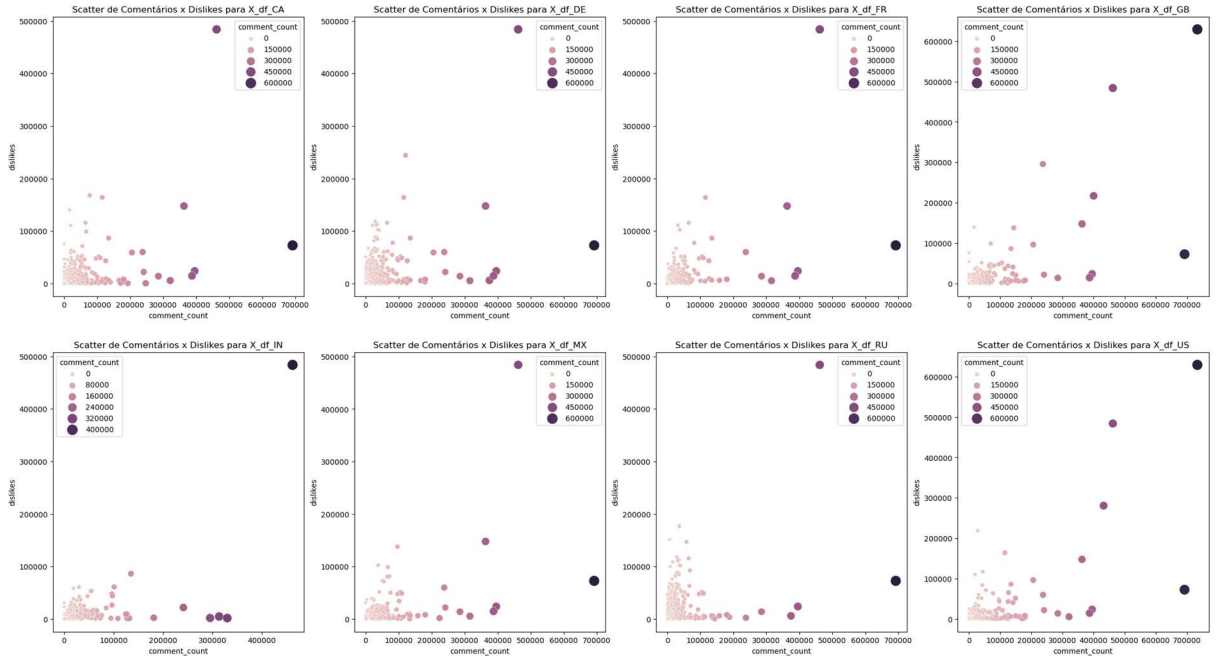
Analisando os diagramas de dispersão podemos verificar que existe uma certa correlação positiva entre comentários e *likes*, e entre *views* e *likes*. Ou seja, quanto maior a quantidade de *likes*, maior será a quantidade de comentários e visualizações. As demais variáveis demonstraram uma correlação neutra.

Figura 10: Código-Fonte e Scatterplot de Comentários x *Dislikes*

```
#scatter dislikes e comments
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(28, 15))

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    sns.scatterplot(data=df, x='comment_count', y='dislikes', size='comment_count', hue='comment_count', sizes=(20, 200), ax=ax)
    ax.set_title(f'Scatter de Comentários x Dislikes para {nome}')

plt.show()
```



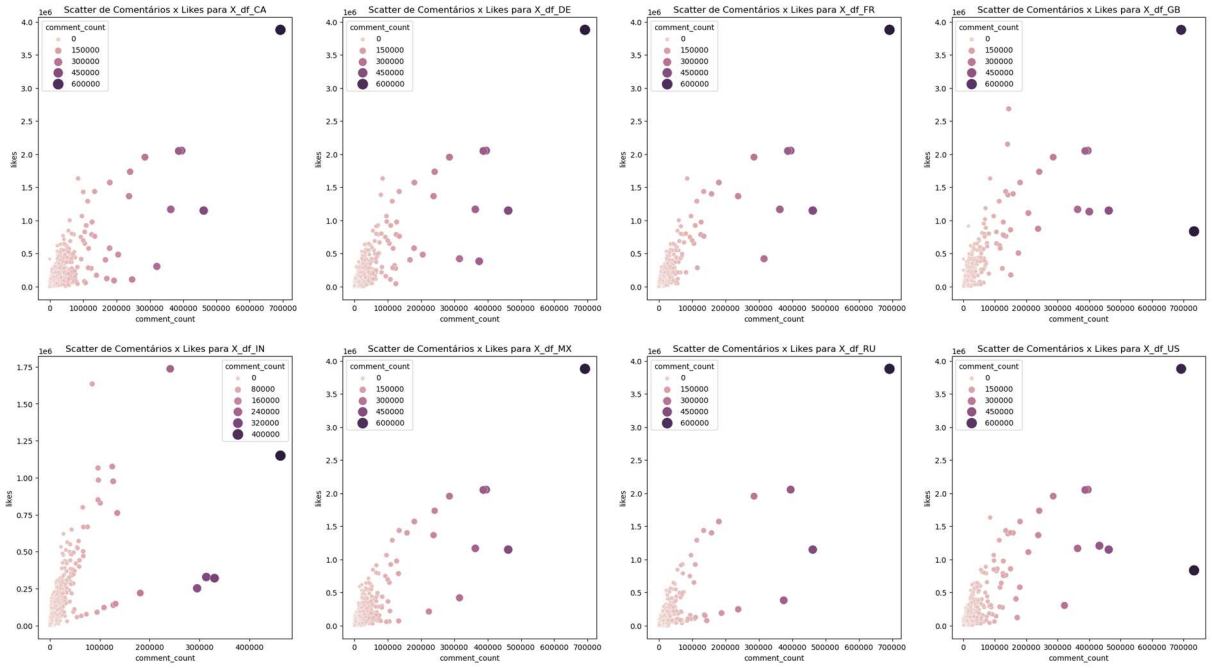
Fonte: Elaborado pelo autor

Figura 11: Scatterplot de Comentários x *Likes*

```
#scatter likes e comments
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(28, 15))

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    sns.scatterplot(data=df, x='comment_count', y='likes', size='comment_count', hue='comment_count', sizes=(20, 200), ax=ax)
    ax.set_title(f'Scatter de Comentários x Likes para {nome}')

plt.show()
```



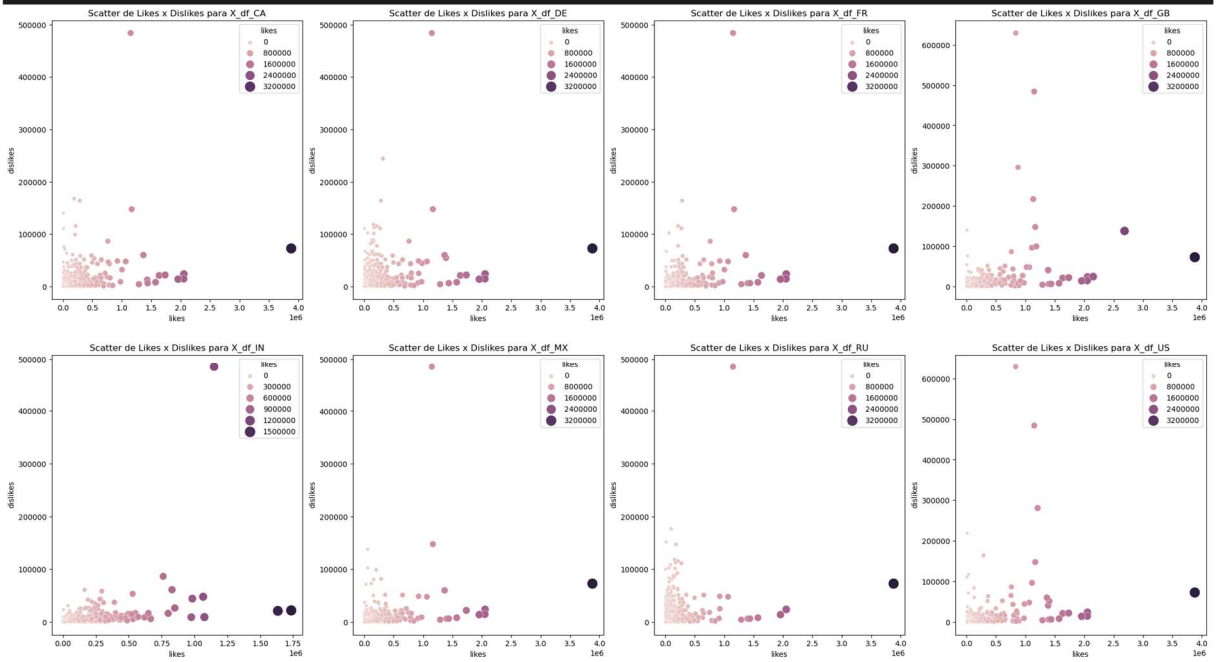
Fonte: Elaborado pelo autor

Figura 12: Scatterplot de *Likes* x *Dislikes*

```
#scatter likes X dislikes
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(28, 15))

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    sns.scatterplot(data=df, x='likes', y='dislikes', size='likes', hue='likes', sizes=(20, 200), ax=ax)
    ax.set_title(f'Scatter de Likes x Dislikes para {nome}')

plt.show()
```



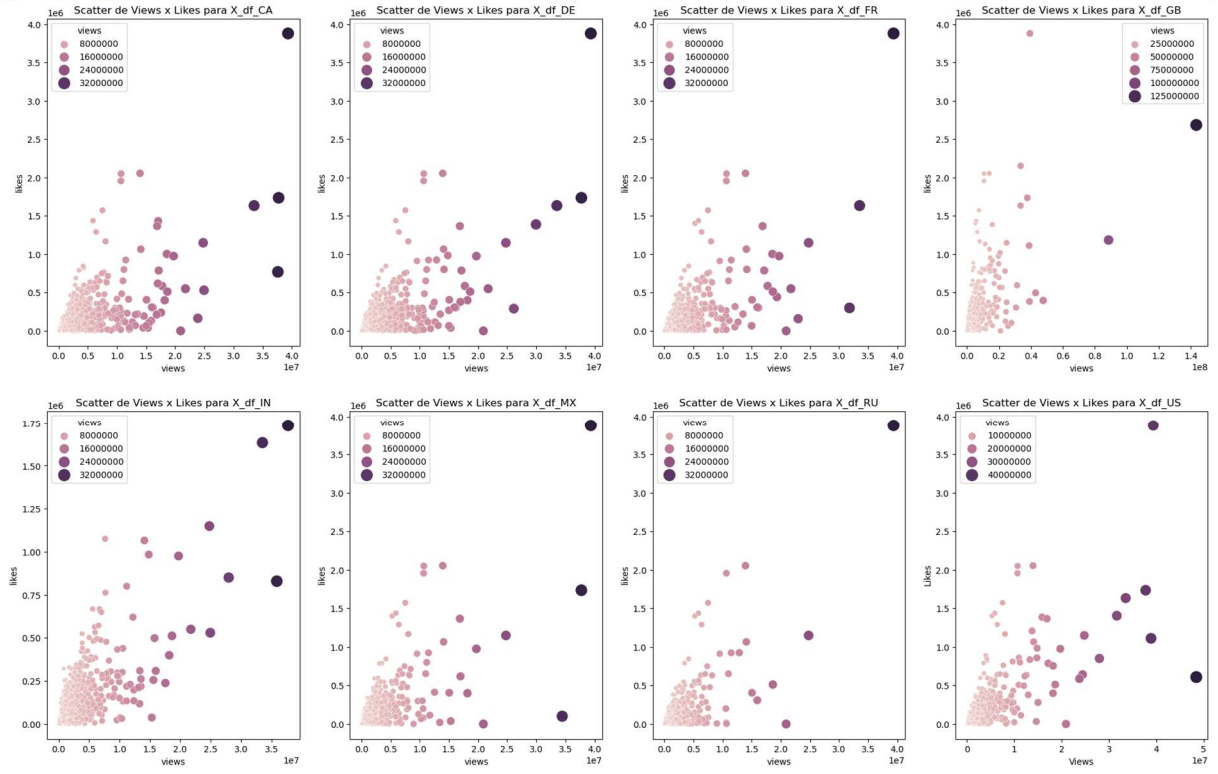
Fonte: Elaborado pelo autor

Figura 13: *Scatterplot* de Visualizações x *Likes*

```
#scatter views x likes
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(24, 15))

for ax, (nome, df) in zip(axes.flat, tabelas.items()):
    df = df.sample(3000)
    sns.scatterplot(data=df, x='views', y='likes', size='views', hue='views', sizes=(20, 200), ax=ax)
    ax.set_title(f'Scatter de Views x Likes para {nome}')

plt.show()
```



Fonte: Elaborado pelo autor

Figura 14: *Scatterplot* de Visualizações x *Dislikes*

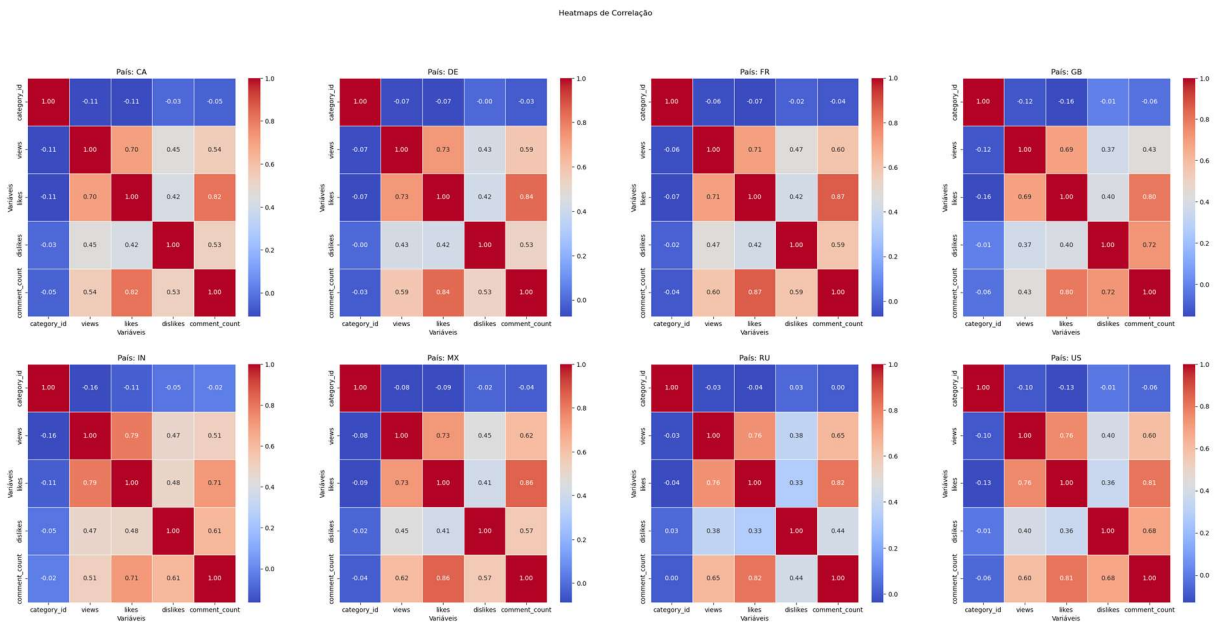
Fonte: Elaborado pelo autor

Na Figura 14 são exibidos os mapas de calor dos 8 países demonstrando a correlação entre as variáveis numéricas.

A maior correlação, presente em 7 dos 8 países, é entre *likes* e comentários com uma média de 0,83. A segunda maior correlação é entre *views* e *likes* com uma média 0,74.

Figura 15: *Heatmap* de dados numéricos

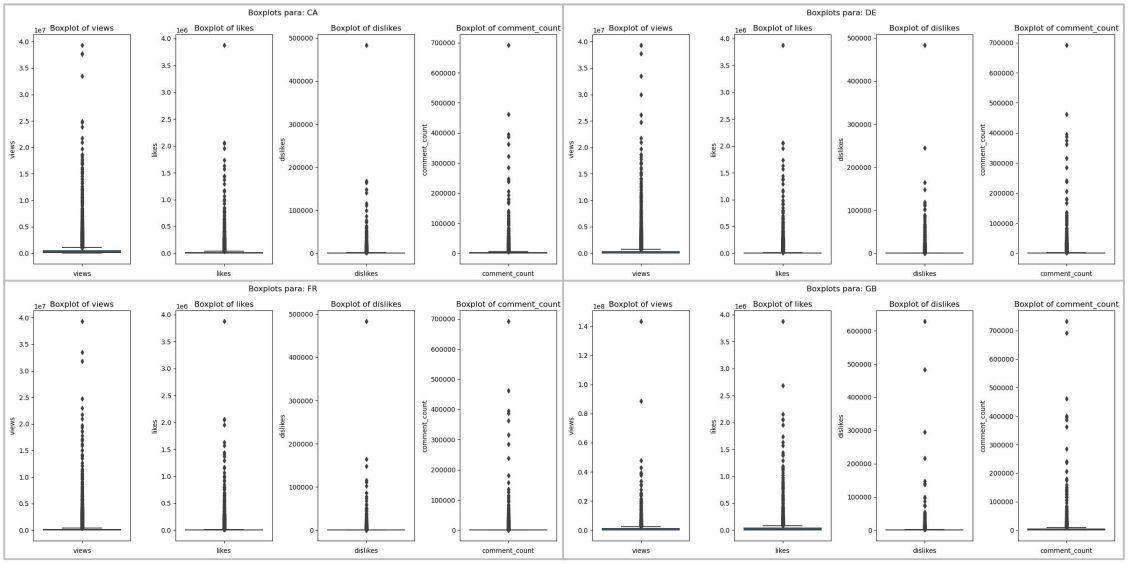
```
#heatmap
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(34, 15))
for ax, (nome, df) in zip(axes.flat, tabelas.items()):
# for nome, df in tabelas.items():
    if df.select_dtypes(include=[np.number]).empty:
        continue
    correlation_matrix = df.select_dtypes(include=[np.number]).corr()
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=.5, ax=ax)
    ax.set_title(f'País: {nome[-2:]}')
    ax.set_xlabel('Variáveis')
    ax.set_ylabel('Variáveis')
plt.suptitle('Heatmaps de Correlação')
plt.show()
```



Fonte: Elaborado pelo autor

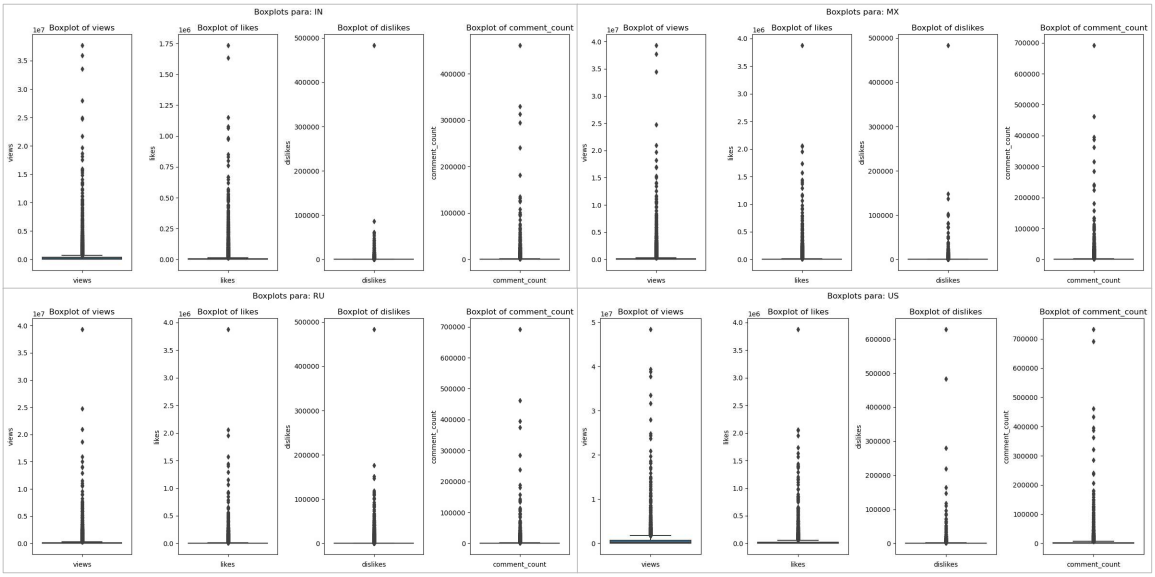
As Figuras 16 e 17 demonstram os *boxplots* de cada país com a distribuição de visualizações, *likes*, *dislikes* e comentários. Porém, por se tratar dos dados originais e conter muitas amostras não é possível distinguir a mediana e os quartis, mas os *outliers* já ficam evidentes. Na Figura 17 é apresentado o código-fonte utilizado para gerar esses gráficos.

Figura 16: *Boxplots* de *views*, *likes*, *dislikes* e comentários de Canadá, Dinamarca, França e Grã-Bretanha



Fonte: Elaborado pelo autor

Figura 17: *Boxplots* de *views*, *likes*, *dislikes* e comentários de Índia, México, Rússia e Estados Unidos



Fonte: Elaborado pelo autor

Figura 18: Código-fonte para elaboração dos gráficos *boxplot*

```
#boxplot
columns_to_plot = ['views', 'likes', 'dislikes', 'comment_count']
for name, df in tabelas.items():
    plt.figure(figsize=(12, 6))

    for i, column in enumerate(columns_to_plot):
        if column in df.columns:
            plt.subplot(1, len(columns_to_plot), i + 1)
            sns.boxplot(y=df[column])
            plt.title(f'Boxplot of {column}')
            plt.xlabel(column)
        else:
            print(f"A coluna {column} não existe no DataFrame {name}")

    plt.suptitle(f"Boxplots para: {name[-2:]}")
    plt.tight_layout()
    plt.show()
```

Fonte: Elaborado pelo autor

A Figura 19 apresenta o código-fonte contendo os modelos de predição utilizados. Para o treinamento do algoritmo, foi assumida como variável alvo a quantidade de *views* dos vídeos, ou seja, o problema a ser resolvido está relacionado a utilizar os dados de vídeos (comentários, *likes* e *dislikes*) para prever a quantidade de visualizações. Para validar esse funcionamento, a partir da execução do treinamento, os modelos foram avaliados utilizando a validação cruzada, que é uma técnica usada para avaliar a performance de modelos de aprendizado de máquina, garantindo que eles generalizem bem para dados não vistos. Nessa validação são utilizados como parâmetros o Erro Quadrático Médio (MSE) e Coeficiente de Determinação (R^2).

Figura 19: Código-Fonte do desenvolvimento da análise de dados

```
models = {
    'Linear Regression': LinearRegression(),
    'SVR': SVR(kernel='rbf'),
    'KNN': KNeighborsRegressor(n_neighbors=5),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'Gradient Boosting': GradientBoostingRegressor(n_estimators=100)
}
num_columns = ['views', 'likes', 'dislikes', 'comment_count']
for name, model in models.items():
    for n, t in tabelas.items():
        if all(col in t.columns for col in num_columns):
            X_scaled = t[num_columns]
            y = t['views']  # Assumindo que 'views' é a variável alvo
            X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
            model.fit(X_train, y_train)
            y_pred = model.predict(X_test)
            mse = mean_squared_error(y_test, y_pred)
            scores = cross_val_score(model, X_scaled, y, cv=5, scoring='r2')
            print(f"{name} com dados {n} - MSE: {mse:.4f}, Validação Cruzada R²: {np.mean(scores):.4f} ± {np.std(scores):.4f}")
        else:
            print(f"Faltando colunas necessárias em {n} para o modelo {name}")
```

Fonte: Elaborado pelo autor

Para o treinamento dos modelos, conforme pode ser observado na Figura 19, foram utilizados 20% dos dados disponíveis para cada um dos *datasets* dos países.

A média dos resultados obtidos a partir da análise dos dados foram consolidados na Tabela 1, que apresenta os valores de Erro Quadrático Médio (MSE) e do Coeficiente de Determinação (R^2) para os modelos aplicados.

Tabela 2: Cálculo MSE e R^2 com dados originais

Modelos	**R^2**	**MSE**
Decision Tree	0,9666	0,0456
Gradient Boosting	0,9851	0,0187
KNN	0,9762	0,0345
Linear Regression	1,0000	0,0000
Random Forest	0,9793	0,0246
SVR	0,0581	0,0280
Médias	**0,8082**	**0,0252**

Fonte: Elaborado pelo Autor

Valores menores de MSE indicam um modelo mais preciso. O modelo *Linear Regression* apresenta um valor de MSE de 0, o que geralmente é um indicativo de que pode ter ocorrido um erro na análise ou algum tipo de *overfitting* extremo, visto que não é comum na prática obter um MSE exatamente igual a zero.

Valores mais próximos de 1 indicam um melhor ajuste do modelo. Um R^2 de 1,0000 para *Linear Regression* sugere um ajuste perfeito, o que é inusitado e pode sugerir um *overfitting* severo ou uma anomalia nos dados.

O modelo *Gradient Boosting* é o mais recomendado devido ao seu baixo MSE, alto R^2 e consistência. Os modelos KNN e *Random Forest* também são bons, mas com maior variabilidade e MSE. *Decision Tree* é razoável, mas não o melhor entre os modelos utilizados. SVR não é adequado para este conjunto de dados.

Para colocar em teste os modelos apresentados, foi elaborado um algoritmo, conforme apresentado no Quadro 1, em que foram realizados os seguintes procedimentos:

1. Leitura dos dados
2. Remoção do valor de *views* da última linha dos *dataframes* com os dados lidos
3. Realização da Previsão dos dados de *views* utilizando os diferentes modelos de ML.

Quadro 1: Algoritmo para previsão de dados de *views* em vídeos do *Youtube*

```
import os
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import mean_squared_error
import seaborn as sns
import matplotlib.pyplot as plt

pasta = './data'
files = [f for f in os.listdir(pasta) if f.endswith('.csv')]

dataframes = {}

def read_file(file_path):
    encondings = ['utf-8', 'latin1', 'iso-8859-1']
    for enc in encondings:
        try:
            return pd.read_csv(file_path, encoding=enc)
        except UnicodeDecodeError:
            continue

for i, arq in enumerate(files):
```

```
    file_path = os.path.join(pasta, arq)
    df = read_file(file_path).drop_duplicates(subset=['video_id'])
    df['country'] = arq[:2]
    if df is not None:
        var_name = f'df_{arq[:2]}'
        dataframes[var_name] = df

tabelas = {}
ys = {}

for v_name, data in dataframes.items():
    # Separando valores e Filtrando colunas
    X = data[['views', 'likes', 'dislikes', 'comment_count']].copy()
    titles = data[['video_id', 'trending_date', 'publish_time', 'title', 'category_id', 'country']].copy()

    # Juntando os dataframes
    X = titles.join(X)

    var_name = f'X_{v_name}'
    ys[var_name] = data['views']
    tabelas[var_name] = X

# Remover o último valor e previsionar
models = {
    'Linear Regression': LinearRegression(),
    'SVR': SVR(kernel='rbf'),
    'KNN': KNeighborsRegressor(n_neighbors=5),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'Gradient Boosting': GradientBoostingRegressor(n_estimators=100)
}

num_columns = ['views', 'likes', 'dislikes', 'comment_count']
results = []

for name, model in models.items():
    for n, t in tabelas.items():
        if all(col in t.columns for col in num_columns):
            X_scaled = t[num_columns].copy()
            y = t['views'].copy()

            # Remover o último valor
            X_train = X_scaled.iloc[:-1]
            y_train = y.iloc[:-1]
            X_test = X_scaled.iloc[-1:].copy()
            y_test = y.iloc[-1:].copy()

            # Treinar o modelo e prever o valor removido
            model.fit(X_train, y_train)
            y_pred = model.predict(X_test)
```

```
        mse = mean_squared_error(y_test, y_pred)
        r2_scores = cross_val_score(model, X_train, y_train, cv=5, scoring='r2')

        results.append({
            'Model': name,
            'Dataset': n,
            'Actual Value': y_test.values[0],
            'Predicted Value': y_pred[0]
        })

# Imprimir resultados
for result in results:
    print(f"{result['Model']} com dados {result['Dataset']} - Valor Real: {result['Actual Value']}, Valor Previsto:
{result['Predicted Value']}")
```

Fonte: Elaborado pelo autor

O algoritmo proposto teve um tempo de execução de 33 minutos sendo executado no ambiente *Google Colaboratory.Os* resultados das previsões são apresentados na Tabela 3.

Tabela 3: Resultado das previsões de quantidade de *views* dos vídeos do Youtube

Modelo	Valor Real	Valor Previsto	Diferença
Linear Regression - dataset com dados de vídeos da Rússia	52340	52339,9999	1,01863E-10
Linear Regression - dataset com dados de vídeos da Índia	770873	770873,0000	0
Linear Regression - dataset com dados de vídeos do Canadá	107392	107391,99999	1,00408E-09
Linear Regression - dataset com dados de vídeos do México	108936	108936,0000	0
Linear Regression - dataset com dados de vídeos dos Estados Unidos	296295	296294,99999	1,8953E-10

Linear Regression - dataset com dados de vídeos da França	46604	46604,0000	0
Linear Regression - dataset com dados de vídeos do Reino Unido	772049	772049,0000	0
Linear Regression - dataset com dados de vídeos da Dinamarca	316328	316328,0000	0
SVR - dataset com dados de vídeos da Rússia	52340	52109,0570	230,9429763
SVR - dataset com dados de vídeos da Índia	770873	153696,2272	617176,7728
SVR - dataset com dados de vídeos do Canadá	107392	220032,8686	-112640,8686
SVR - dataset com dados de vídeos do México	108936	42300,7855	66635,21449
SVR - dataset com dados de vídeos dos Estados Unidos	296295	270902,4910	25392,50901
SVR - dataset com dados de vídeos da França	46604	44790,2544	1813,745565
SVR - dataset com dados de vídeos do Reino Unido	772049	278630,7779	493418,2221
SVR - dataset com dados de vídeos da Dinamarca	316328	73392,4225	242935,5775
KNN - dataset com dados de vídeos da Rússia	52340	52567,6000	-227,6
KNN - dataset com dados de vídeos da Índia	770873	771173,2000	-300,2
KNN - dataset com dados de vídeos do Canadá	107392	107402,4000	-10,4

KNN - dataset com dados de vídeos do México	108936	108559,0000	377
KNN - dataset com dados de vídeos dos Estados Unidos	296295	296355,6000	-60,6
KNN - dataset com dados de vídeos da França	46604	46521,0000	83
KNN - dataset com dados de vídeos do Reino Unido	772049	770857,8000	1191,2
KNN - dataset com dados de vídeos da Dinamarca	316328	315349,6000	978,4
Decision Tree - dataset com dados de vídeos da Rússia	52340	52339,0000	1
Decision Tree - dataset com dados de vídeos da Índia	770873	770445,0000	428
Decision Tree - dataset com dados de vídeos do Canadá	107392	107448,0000	-56
Decision Tree - dataset com dados de vídeos do México	108936	108933,0000	3
Decision Tree - dataset com dados de vídeos dos Estados Unidos	296295	296237,0000	58
Decision Tree - dataset com dados de vídeos da França	46604	46619,0000	-15
Decision Tree - dataset com dados de vídeos do Reino Unido	772049	771567,0000	482
Decision Tree - dataset com dados de vídeos da Dinamarca	316328	316455,0000	-127

Random Forest - dataset com dados de vídeos da Rússia	52340	52339,2300	0,77
Random Forest - dataset com dados de vídeos da Índia	770873	770761,3200	111,68
Random Forest - dataset com dados de vídeos do Canadá	107392	107392,1300	-0,13
Random Forest - dataset com dados de vídeos do México	108936	108952,3200	-16,32
Random Forest - dataset com dados de vídeos dos Estados Unidos	296295	296299,6800	-4,68
Random Forest - dataset com dados de vídeos da França	46604	46602,1600	1,84
Random Forest - dataset com dados de vídeos do Reino Unido	772049	772535,1700	-486,17
Random Forest - dataset com dados de vídeos da Dinamarca	316328	316468,5000	-140,5
Gradient Boosting - dataset com dados de vídeos da Rússia	52340	53504,3146	-1164,314646
Gradient Boosting - dataset com dados de vídeos da Índia	770873	787748,0085	-16875,00848
Gradient Boosting - dataset com dados de vídeos do Canadá	107392	106208,3627	1183,637288

Gradient Boosting - dataset com dados de vídeos do México	108936	109658,7590	-722,7589529
Gradient Boosting - dataset com dados de vídeos dos Estados Unidos	296295	293710,7848	2584,215191
Gradient Boosting - dataset com dados de vídeos da França	46604	46733,0648	-129,0647993
Gradient Boosting - dataset com dados de vídeos do Reino Unido	772049	765319,4980	6729,501979
Gradient Boosting - dataset com dados de vídeos da Dinamarca	316328	333413,1335	-17085,13355

Fonte: Elaborado pelo autor

Conforme realçado na Tabela 2, o modelo de Regressão Linear foi o modelo que mais se aproximou do valor real da quantidade de *views* dos vídeos, seguido do modelo Random Forest. Já o pior modelo utilizado foi o SVR, onde valores muito discrepantes foram previstos.

Esses resultados demonstram uma situação que pode servir de base para outros profissionais na área de Ciência de Dados como proceder para realizar a avaliação, testagem e produção a partir de algoritmos de Aprendizagem de Máquina.

7 CONCLUSÃO

Este estudo demonstra o potencial do *Machine Learning* na análise de dados de vídeos do *YouTube*, fornecendo uma base sólida para futuras pesquisas e aplicações práticas no campo do marketing digital e criação de conteúdo. A combinação de diferentes modelos permite uma abordagem muito ampla que pode capturar tanto aspectos quantitativos quanto qualitativos do desempenho dos vídeos, bem como a identificação de padrões e tendencias.

Futuros estudos podem explorar a integração de mais fontes de dados utilizando outras plataformas de mídia social para uma análise mais abrangente, experimentar a utilização de redes neurais e modelos de *deep learning* mais avançados para melhorar os resultados e desenvolver sistemas que possam analisar e prever o desempenho de vídeos em tempo real, oferecendo feedback imediato para os criadores de conteúdo.

REFERÊNCIAS

BARRETT, James P. The coefficient of determination—some limitations. **The American Statistician**, v. 28, n. 1, p. 19-20, 1974.

BREIMAN, L. Random forests. **Machine learning**, v. 45, n. 1, p. 5-32, 2001.

CUMBI, Eurico Fernandes José. **Tratamento de dados com Outliers: uma aplicação à análise de Regressão**. 2010. 54 p. Trabalho de Conclusão de Curso. 2010. Universidade Eduardo Mondlane. Licenciatura em Estatística. 2010.

DAVIS, J.; GOADRICH, M. The relationship between precision-recall and ROC curves. In: **Proceedings of the 23rd international conference on Machine learning** (ICML). 2006. p. 233-240.

DRAPER, Norman R.; SMITH, Harry. **Applied regression analysis**. John Wiley & Sons, 1998.

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern Classification**. 2. ed. Wiley, 2012.

FREITAS, André Lage; SANTANA JR, Orivaldo Vieira. Machine Learning: desafios para um Brasil competitivo**. Computação Brasil**, n. 39, p. 7-10, 2019.

HOLMES, W.; BIALIK, M.; FADEL, C. **Artificial Intelligence in Education – Promises and Implications for Teaching and Learning**. Boston: The Center for Curriculum Redesign, 2019.

JOHNSON, Richard Arnold et al. **Applied multivariate statistical analysis**. 2002.

JORDAN, M. I.; MITCHELL, T. M. Machine learning: Trends, perspectives, and prospects. **Science**, v. 349, n. 6245, p. 255-260, 2015.

KOLLMANNSBERGER, Stefan et al. **Deep learning in computational mechanics**. Springer International Publishing, 2021.

LIAW, Andy et al. Classification and regression by randomForest. **R news**, v. 2, n. 3, p. 18-22, 2002.

MITCHELL, T. M. **Machine Learning**. McGraw-Hill, 1997.

NGUYEN, Dong et al. Joint network coding and machine learning for error-prone wireless broadcast. In: **2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)**. IEEE, 2017. p. 1-7.

POWERS, D. M. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. **Journal of Machine Learning Technologies**, v. 2, n. 1, p. 37-63, 2011.

RUSSELL, S.; NORVIG, P. **Inteligência Artificial**. 3. ed. [S.l.]: Elsevier, 2013.

SATHYA, R.; ABRAHAM, A. Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification. **International Journal of Artificial Intelligence**, v. 2, n. 2, p. 34-8, 2013.

SCHLUCHTER, Mark D. Mean square error. **Encyclopedia of Biostatistics**, v. 5, 2005.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Information Processing & Management**, v. 45, n. 4, p. 427-437, 2009.