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

RAFAEL DE SOUZA EMIDIO DA SILVA

GAME DATA SCIENCE:
GERAÇÃO DE INFERÊNCIAS COM INTELIGÊNCIA ARTIFICIAL UTILIZANDO DADOS GERADOS POR JOGOS DIGITAIS.

Sorocaba
2025

RAFAEL DE SOUZA EMIDIO DA SILVA

GAME DATA SCIENCE:

GERAÇÃO DE INFERÊNCIAS COM INTELIGÊNCIA ARTIFICIAL UTILIZANDO DADOS GERADOS POR JOGOS DIGITAIS

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

Orientador(a): Prof. Dr. Leopoldo André Dutra Lusquino Filho

Sorocaba

2025

S586g

Silva, Rafael de Souza Emidio da
Game Data Science : geração de inferências com Inteligência Artificial utilizando dados gerados por jogos digitais / Rafael de Souza Emidio da Silva. -- Sorocaba, 2025
55 p. : il., tabs.

Trabalho de conclusão de curso (Bacharelado - Engenharia de Controle e Automação) - Universidade Estadual Paulista (UNESP), Instituto de Ciência e Tecnologia, Sorocaba
Orientador: Leopoldo André Dutra Lusquino Filho

1. Aprendizado do computador. 2. Análise de regressão. 3. Árvores de decisão. 4. Jogos para computador. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

RAFAEL DE SOUZA EMIDIO DA SILVA

GAME DATA SCIENCE:
GERAÇÃO DE INFERÊNCIAS COM INTELIGÊNCIA ARTIFICIAL UTILIZANDO DADOS GERADOS POR JOGOS DIGITAIS

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

Data da defesa: 20/05/2025

BANCA EXAMINADORA:

Prof. Dr. Leopoldo André Dutra Lusquino Filho
UNESP – Instituto de Ciência e Tecnologia – Campus de Sorocaba

Prof. Dr. Eduardo Verri Liberado
UNESP – Instituto de Ciência e Tecnologia – Campus de Sorocaba

Sidney Alves de Outeiro
UFRJ – Universidade Federal do Rio de Janeiro

Aos meus pais, Iani e Paulo, por sempre cuidarem de mim e do meu irmão da melhor forma que um filho pode imaginar, pelos conselhos, broncas, por sempre nos inspirarem a sermos pessoas melhores e por sempre fazerem o possível e o impossível para que tivéssemos uma formação de qualidade. Ao meu irmão, Diego, por todos esses anos que passamos juntos vivendo literalmente no mesmo quarto. Sua companhia e amizade foram essenciais para que eu pudesse estar onde estou hoje. À minha namorada, Gabriela, o amor da minha vida, por ser essa pessoa incrível, de quem eu me orgulho muito e sempre me apoiou durante essa jornada. A vida tem mais graça com você.

AGRADECIMENTOS

Em primeiro lugar gostaria de agradecer aos meus pais, Iani e Paulo, pelo amor incondicional, apoio constante, pela fé em mim mesmo nos momentos mais difíceis e por nunca terem medido esforços para me proporcionar um ensino de qualidade durante todo o meu período escolar.

À minha namorada, Gabriela, por me dar forças, me apoiar, por dividir tudo comigo e sempre estar ao meu lado.

Ao meu irmão Diego, pela parceria, brincadeiras e companheirismo durante essa jornada.

À minha família e meus amigos por sempre torcerem por mim, pelos conselhos valiosos e pela troca de experiências que me permitiram crescer não só como pessoa, mas também como formando.

Aos colegas da minha turma, XVII, das turmas XVI, XV e XIV do curso de Engenharia de Controle a Automação, pela amizade, pela parceira, pelas brincadeiras e pelo apoio durante a trajetória do curso.

Ao ISMART, por acreditar em mim e investir nos meus estudos, permitindo que eu tivesse uma formação de excelência.

Ao meu orientador, Prof. Dr. Leopoldo André Dutra Lusquino Filho, pela orientação, conselhos, dicas e direcionamentos durante este trabalho.

Essa caminhada foi marcada por aprendizados que levarei para toda a vida. A todos que, de alguma forma, estiveram ao meu lado, meus mais sinceros agradecimentos.

“Wish we could turn back time to the good old days
When our mama sang us to sleep, but now we're stressed out.”
(Twenty One Pilots, 2015).

RESUMO

Este estudo teve como objetivo explorar a aplicação de técnicas estatísticas e de inteligência artificial na extração de inferências a partir de dados de partidas do jogo League of Legends, com ênfase na análise do desempenho de jogadores e equipes em ambientes competitivos. Utilizando dados públicos da API da Riot Games, foram coletadas e analisadas estatísticas de quatro jogadores profissionais brasileiros, aplicando métodos estatísticos descritivos e testes de hipótese para identificar padrões relevantes. Em seguida, algoritmos de aprendizado de máquina como Random Forest, XGBoost e SVM foram utilizados para tarefas de regressão e classificação, avaliando a capacidade preditiva dos modelos por meio de métricas como MAE, RMSE, acurácia, recall e F1-score. O SVM se destacou na previsão de vitórias, atingindo até 80,8% de recall. O estudo conclui que, mesmo com limitações como tamanho reduzido da amostra e ausência de variáveis contextuais, é possível extrair insights significativos de partidas com potencial para aplicação em estratégias de equipes, desenvolvimento de ferramentas analíticas e suporte a decisões no cenário de *e-sports*.

Palavras-chave: inteligência artificial; estatística aplicada; League of Legends; análise de dados; aprendizado de máquina; métricas de performance; árvores de decisão.

ABSTRACT

This study aimed to explore the application of statistical techniques and artificial intelligence to generate inferences from match data in the game League of Legends, with an emphasis on analyzing the performance of players and teams in competitive environments. Using public data from the Riot Games API, statistics from four professional Brazilian players were collected and analyzed, applying descriptive statistical methods and hypothesis testing to identify relevant patterns. Subsequently, machine learning algorithms such as Random Forest, XGBoost, and SVM were applied to regression and classification tasks, evaluating the predictive capacity of the models using metrics such as MAE, RMSE, accuracy, recall, and F1-score. SVM stood out in victory prediction, achieving up to 80.8% recall. The study concludes that, despite limitations such as a small sample size and lack of contextual variables, it is possible to extract meaningful insights from matches with potential applications in team strategies, the development of analytical tools, and decision support in the e-sports scenario.

Keywords: artificial intelligence; applied statistics; League of Legends; data analysis; machine learning; performance metrics; decision trees.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

A Inteligência Artificial (IA) tem desempenhado um papel cada vez mais relevante no desenvolvimento de jogos digitais, possibilitando a criação de experiências mais imersivas e dinâmicas. Além de aprimorar mecânicas de jogo e a interação com *NPC's,* a IA também permite a análise de grandes volumes de dados gerados durante as partidas (Sapienza; Bessi; Ferrara. 2017). Para que essas informações sejam utilizadas de forma eficiente, a análise estatística se torna uma ferramenta fundamental, ajudando a identificar padrões, prever comportamentos e fornecer suporte para a tomada de decisões dentro do jogo.

Esse tipo de aprimoramento dos jogos digitais é um dos principais fatores que mantêm esse mercado em constante crescimento ao redor do mundo. Praticamente toda tecnologia recém-desenvolvida acaba sendo testada na indústria de games, impulsionando avanços em inteligência artificial, gráficos, redes e processamento de dados, que beneficiam desenvolvedores com ferramentas mais sofisticadas e jogadores com experiências mais imersivas. Como reflexo desse impacto, a indústria de jogos digitais gerou US$ 175,8 bilhões em 2021, consolidando-se como uma das mais lucrativas do setor de entretenimento (Newzoo, 2021). Além do impacto tecnológico e financeiro dos games, outro fator importante para esse crescimento é a popularização das competições de video-games, onde teve origem o termo "*E-sports*". A modalidade esportiva permite à indústria de jogos digitais ter uma participação no mundo dos esportes, movimentando times profissionais, transmissões ao vivo, patrocinadores entre outros, podendo coletar uma parte dessa arrecadação (Data Insights, 2022). A título de comparação com modalidades esportivas convencionais, em 2015, um total de 36 milhões de pessoas assistiram às finais do Campeonato Mundial de League of Legends, contra cerca de 20 milhões de espectadores no jogo número seis das finais da liga de basquete americana NBA (UOL, 2015), evento que sempre reúne milhões de fãs ao redor do mundo todo.

O uso de dados coletados por meio de APIs e bancos públicos possibilita a aplicação de técnicas estatísticas e de aprendizado de máquina, permitindo a extração de informações sobre desempenho, padrões e comportamentos nas partidas. A programação em Python, aliada a bibliotecas como NumPy e Pandas, viabiliza o tratamento, organização e transformação desses dados, garantindo uma base consistente para a construção e avaliação dos modelos preditivos. A partir dessa abordagem, é possível explorar como a IA e a estatística podem ser utilizadas para entender tendências, otimizar mecânicas e aprimorar a personalização da experiência nos jogos digitais.

No entanto, essa abordagem também impõe uma série de desafios técnicos e analíticos que precisam ser superados para que os resultados sejam significativos, desde a complexidade

no tratamento dos dados até a validação das inferências geradas. Jogos modernos produzem enormes quantidades de dados (logs, interações, estatísticas das partidas, entre outras). Essa coleta de informações, principalmente no cenário competitivo onde cada detalhe pode fazer uma grande diferença, pode trazer insights importantes e valiosos para que estratégias sejam tomadas pelas equipes e pelos players. De modo a minimizar erros, elaborar estratégias mais sólidas baseadas nas informações coletadas é primordial hoje em dia para prever resultados, comportamentos e maximizar suas chances de vitória.

Entre os desafios, destaca-se a dificuldade em obter dados relevantes e atualizados de forma consistente, especialmente quando a coleta depende de APIs ou serviços externos sujeitos a limitações e instabilidades. Também aplicar modelos eficientes independentes de processamento em nuvem pode limitar os recursos computacionais disponíveis (Chamma *et al*., 2021). Para garantir a identificação de padrões úteis e significativos, uma etapa primordial é a filtragem de dados relevantes e informações que realmente gerem valor para que seja feita uma inferência relevante.

Diante desse contexto, este trabalho tem como foco investigar de que maneira métodos estatísticos e algoritmos de aprendizado de máquina podem ser aplicados à análise de dados de partidas do jogo League of Legends, com o objetivo de gerar inferências sobre desempenho e resultado das equipes. O problema de pesquisa concentra-se na aplicação dessas técnicas para compreender se é possível extrair informações relevantes a partir de dados públicos que permitam identificar padrões e auxiliar na interpretação e previsão de comportamentos no ambiente competitivo do jogo.

Neste trabalho, o capítulo 2 apresenta os trabalhos relacionados e as abordagens existentes na literatura sobre estatística e aprendizado de máquina aplicados a jogos digitais. O capítulo 3 descreve a metodologia adotada, incluindo os critérios para seleção do jogo, o processo de coleta de dados, a escolha dos jogadores de referência e as análises estatísticas realizadas. Em seguida, o capítulo 4 detalha o conjunto de dados utilizado, os resultados da análise estatística e os testes realizados com diferentes algoritmos de machine learning, abordando tanto regressão quanto classificação. Por fim, o capítulo 5 traz as considerações finais do estudo, com um resumo dos principais resultados, limitações encontradas e sugestões de trabalhos futuros.

2. TRABALHOS RELACIONADOS

2.1. Estatística para jogos

O uso da estatística em jogos digitais tem se mostrado essencial para compreender o comportamento dos jogadores e promover a melhoria contínua da experiência de jogo (Qiu, Gong, Liu, 2024). Ferramentas estatísticas auxiliam na identificação de padrões de desempenho, avaliação de estratégias e análise da dinâmica entre equipes. Em jogos como League of Legends, onde cada partida gera uma grande quantidade de dados — como número de abates, mortes, assistências, ouro acumulado e controle de objetivos —, é possível analisar a influência de cada fator no resultado da partida.

Estudos anteriores como o de Semenov *et al* (2016) utilizaram métodos como análise de variância, regressões lineares e estatísticas descritivas para investigar quais variáveis mais impactam o desempenho de jogadores e times em cenários competitivos. Além disso, é comum o uso de distribuições, medidas de tendência central, dispersão, correlação e testes de hipótese para identificar padrões em grandes volumes de dados. A seguir, são apresentados alguns exemplos relevantes de aplicação da estatística e análise de dados no estudo do comportamento de jogadores:

a. Análise de padrões comportamentais em League of Legends

Sapienza, Bessi e Ferrara (2017) aplicaram a técnica de Fatoração de Tensor Não Negativa (NTF) sobre aproximadamente 100 mil partidas de League of Legends, com o objetivo de identificar padrões de comportamento e agrupamentos de jogadores com estilos e progressões semelhantes ao longo do tempo. Essa abordagem permitiu compreender como os jogadores aprendem e evoluem suas habilidades dentro do jogo.

b. Análise de comportamento de usuários em um jogo MMO *mobile*

Qiu, Gong e Liu (2024) realizaram uma análise abrangente do comportamento de usuários em um popular jogo mobile do gênero battle royale. Utilizando técnicas de mineração de dados temporais e estáticos, como *clustering K-means* em séries temporais e algoritmos baseados em grafos (DeepWalk e LINE), os pesquisadores identificaram segmentos distintos de jogadores, revelando variações significativas no engajamento, nos níveis de habilidade e nas interações sociais entre os grupos analisados.

2.2. Machine Learning para análise de jogos

Machine Learning (ou aprendizado de máquina) é um campo da inteligência artificial que desenvolve algoritmos capazes de aprender padrões a partir de dados e realizar previsões ou decisões sem serem explicitamente programados para isso. Em vez de seguir regras fixas, os modelos de machine learning ajustam seus parâmetros com base nos exemplos que recebem, tornando-se úteis para tarefas como reconhecimento de padrões, classificação, regressão e detecção de anomalias em diversos contextos (Géron, 2019).

Assim como no trabalho de Sapienza, Bessi e Ferrara (2017), onde foi processado um grande conjunto de dados, o uso de técnicas de aprendizado de máquina tem se tornado cada vez mais comum na análise de jogos digitais, especialmente em contextos competitivos ou com grande volume de dados. Por meio de algoritmos supervisionados e não supervisionados, é possível não apenas entender padrões já existentes, mas também fazer previsões sobre resultados, comportamento de jogadores e tomadas de decisão dentro das partidas (Yang; Harrison; Roberts. 2016)

Uma contribuição relevante na área é a de xPetu (2024), que desenvolveu um modelo baseado em redes neurais profundas para estimar, em tempo real, a probabilidade de vitória em partidas de League of Legends. O autor também propõe uma métrica chamada *win probability added*, que busca avaliar o impacto de ações específicas, como a compra de itens, no desfecho da partida. Embora o foco seja em decisões estratégicas de alto nível, o estudo reforça a viabilidade de inferências automatizadas a partir de dados extraídos diretamente do jogo.

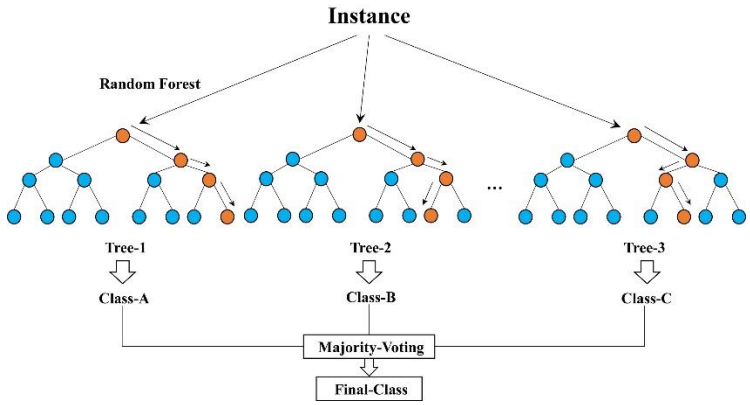
Entre os exemplos de aplicação, destaca-se o estudo de Río, Chen e Periáñez. (2019), que utilizou redes neurais recorrentes (LSTM) para prever o engajamento futuro de jogadores com base em métricas como tempo de jogo e níveis atingidos. Já Semenov *et al*. (2016) aplicaram classificadores como Naïve Bayes e SVM (*Support Vector Machine*) para prever o vencedor de partidas de Dota 2, com base nas composições das equipes e estatísticas iniciais de jogo. Esses exemplos reforçam como modelos de ML podem ser eficazes em ambientes de alto dinamismo e grande volume de dados, características típicas dos jogos online modernos.

2.2.1 Algoritmos Random Forest, XGBoost e SVM

O Random Forest (RF) é um dos algoritmos supervisionados mais amplamente utilizados em problemas de classificação e regressão, especialmente quando se busca robustez, interpretabilidade e boa performance (Durachman; Rahman. 2023). Ele funciona como um modelo de ensemble learning, ou seja, uma combinação de diversos modelos mais simples — no caso, árvores de decisão — treinadas de forma independente a partir de subconjuntos aleatórios dos dados originais (técnica conhecida como bootstrap).

Cada árvore realiza uma predição e, ao final, o modelo consolida todas as saídas por meio de votação majoritária (para classificação) ou média aritmética (para regressão), tornando o processo menos suscetível a *overfitting* e aumentando a precisão geral. Essa abordagem é especialmente útil para jogos digitais, onde os dados são muitas vezes ruidosos, complexos e com múltiplas variáveis. Além disso, o Random Forest permite extrair a importância de cada variável no modelo, o que é valioso para entender quais estatísticas influenciam mais o desempenho dos jogadores ou o resultado das partidas. A Figura 1 apresenta um diagrama simplificado do funcionamento do Random Forest, mostrando como diferentes árvores podem gerar diferentes previsões e como a decisão final é obtida por votação.

Figura 1 - Diagrama do funcionamento do algoritmo Random Forest.



Fonte: Zhu; Zhou; Zhang. 2021

Estudos como o de Yang, Harrison e Roberts (2016) demonstraram a eficácia do algoritmo em prever o resultado de partidas de League of Legends e Heroes of the Storm, utilizando apenas estatísticas iniciais das partidas. Sua capacidade de operar bem com dados offline e históricos o torna ideal para aplicações como a deste trabalho, onde os modelos são treinados localmente sem necessidade de processamento em tempo real.

O trabalho de Durachman e Rahman (2023) investigou a influência do tempo de jogo sobre a recomendação de jogos na plataforma Steam, comparando os modelos de Regressão Logística e Random Forest. Os resultados demonstraram que o Random Forest obteve maior acurácia (de 82,65%) e F1-score na predição das recomendações dos usuários, além de oferecer melhor interpretação sobre a relevância das variáveis utilizadas no modelo.

Semelhante ao Random Forest, mas geralmente considerado mais preciso, o XGBoost (eXtreme Gradient Boosting) é um algoritmo de aprendizado de máquina baseado em árvores de decisão (IBM, 2024). Seu funcionamento se baseia na construção sequencial de estimadores fracos, em que cada nova árvore é treinada para corrigir os erros cometidos pelas anteriores, aprimorando progressivamente a capacidade preditiva do conjunto.

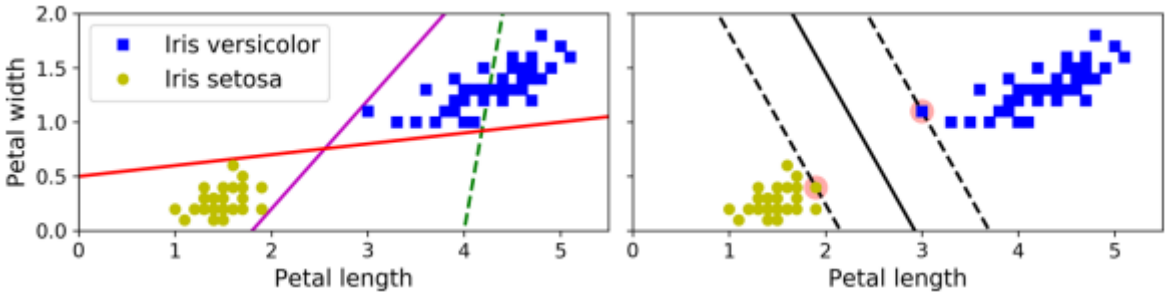
Da sua lógica matemática, sabe-se que:

> Ele expande a função de perda utilizando a fórmula de Taylor de segunda ordem para garantir precisão nos cálculos, removendo os termos constantes e otimizando os termos da função de perda. Além disso, utiliza termos de regularização para evitar o overfitting; esses também são expandidos, os termos constantes são novamente removidos e os componentes de regularização são otimizados. Por fim, os coeficientes dos termos linear e quadrático são combinados para formar a função objetivo final. A estrutura de armazenamento em blocos permite ainda a realização de cálculos em paralelo. (Li; Wang; Zou. 2024. tradução nossa).

Assim como o RF (Random Forest), ele pode ser utilizado tanto para regressão (XGBoost Regressor), onde são analisadas variáveis compostas por números contínuos (Ex.: 0.125), quanto para classificação (XGBoost Classifier), onde são avaliadas variáveis de classificação (como True ou False, por exemplo).

Além dos algoritmos baseados em árvores, também foi utilizado o Support Vector Machine, um algoritmo com uma estratégia distinta, baseada na maximização das margens de separação entre as classes ou, no caso da regressão, na definição de uma faixa ideal de tolerância ao erro (Gerón, 2019). Assim como os anteriores, o SVM também pode ser aplicado tanto em tarefas de regressão (SVR) quanto de classificação (SVC), sendo amplamente reconhecido por sua eficácia em conjuntos de dados com padrões não lineares e alta dimensionalidade. A Figura 2 mostra um exemplo de como funciona essa classificação.

Figura 2 - Ilustração da margem de separação no algoritmo SVM.



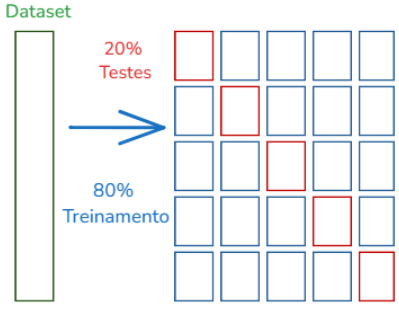
Fonte: Gerón, 2019

A imagem mostra diferentes formas de separar duas espécies de flores com base em suas características. No gráfico da esquerda, três linhas exemplificam formas distintas de como é possível dividir os grupos, mas nem todas fazem isso bem — algumas estão muito próximas dos pontos, o que pode causar erros com novos dados. Já o gráfico da direita mostra como o algoritmo SVM escolhe a melhor linha: aquela que separa os grupos mantendo a maior distância possível dos pontos mais próximos. Essa "margem" maior ajuda o modelo a funcionar melhor com novos casos.

2.2.2 Métricas utilizadas

Para avaliar o desempenho dos modelos de aprendizado de máquina aplicados neste estudo, foram utilizadas métricas distintas para as tarefas de regressão e classificação, de acordo com as práticas consolidadas na literatura. Na implementação dos algoritmos de Machine Learning, uma das técnicas de validação utilizadas foi o K-Fold Cross Validation. Esse método particiona o conjunto de dados em K partes (ou "folds"), utilizando uma delas para validação e as demais para o treinamento do modelo, de forma alternada a cada iteração. (Datacamp, 2024). A Figura 3 ilustra de forma visual o particionamento utilizando um K = 5, por exemplo.

Figura 3 - Exemplo visual do particionamento de dados no K-Fold Cross Validation.



Fonte: Autoria própria.

No caso da regressão, as duas principais métricas empregadas foram o Erro Médio Absoluto (Mean Absolute Error – MAE) e a Raiz do Erro Quadrático Médio (Root Mean Squared Error – RMSE). Ambas permitem quantificar o quão distante, em média, os valores previstos estão dos valores reais.

O MAE mede a média das diferenças absolutas entre os valores previstos e os reais. É uma métrica simples e intuitiva, menos sensível a outliers, e fornece uma noção direta da magnitude média do erro.

$$\mathrm{MAE} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i}=1}^{\mathrm{n}}|\mathrm{y_i} - \widehat{\mathrm{y_i}}| \tag{1}$$

O RMSE, por sua vez, eleva os erros ao quadrado antes de calcular a média, o que faz com que desvios maiores sejam penalizados com mais intensidade. Dessa forma, o RMSE é mais sensível à presença de valores extremos e é útil quando se deseja enfatizar grandes erros de previsão.

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{(y_i - \widehat{y_i})^2}{n}} \tag{2}$$

Para a classificação da variável binária Team Win (vitória ou derrota), foram utilizadas as métricas padrões: acurácia, *precision*, *recall* e *F1-score*. Essas métricas permitiram avaliar o desempenho dos modelos de forma mais completa, levando em consideração tanto a capacidade de acerto geral quanto o equilíbrio entre falsos positivos e falsos negativos.

3. METODOLOGIA

3.1. Critérios para seleção do jogo

Os critérios considerados para a escolha do jogo a ser utilizado no projeto envolveram a facilidade de coleta de dados, a quantidade de informações disponibilizadas e o tipo de jogo. Dessa forma, o jogo selecionado deveria possuir métodos de aquisição de dados acessíveis e volumes expressivos de informações, permitindo que as análises fossem conduzidas de forma mais precisa e robusta. Em um primeiro momento, os games a serem considerados para utilização foram: Brawl Stars (plataformas Android e iOS), Hollow Knight (PC), Terraria (PC), Stardew Valley (PC) e League of Legends (PC).

Com esses critérios em mente, iniciou-se a avaliação das opções disponíveis. De início os jogos em single-player e/ou modo história foram descartados. Em geral, jogos single-player apresentam uma progressão de gameplay mais padronizada e moldada, mesmo que haja alguma liberdade de escolha pelo jogador. Além disso, esses jogos frequentemente salvam os dados localmente, dificultando o acesso a informações de diferentes jogadores. Esse cenário conflitou diretamente tanto com o critério de facilidade de coleta de dados quanto com o de quantidade de dados disponíveis, levando à eliminação dessas opções.

Por outro lado, os jogos online e multiplayer se mostraram mais adequados para os propósitos do projeto. A possibilidade de registrar atributos, resultados e características de cada jogador em plataformas online tornava a coleta de dados mais prática e escalável. Nesse contexto, Brawl Stars e League of Legends se destacaram entre as alternativas.

Entretanto, Brawl Stars é um jogo cuja base de jogadores é majoritariamente composta por público infantil, o que, aliado à baixa disponibilidade de APIs e à ausência de métodos nativos para exportação de dados, comprometeria a aquisição de informações em larga escala necessárias para o estudo. Dessa forma, essa opção também foi descartada.

Entre esses dois, League of Legends destacou-se por possuir uma API robusta e bem documentada, permitindo acesso a uma grande variedade de dados, como estatísticas de partidas, desempenho individual e informações sobre os *players*. Além disso, sua popularidade garantiu um volume significativo de dados disponíveis para análise, atendendo plenamente aos critérios estabelecidos.

3.2. Sobre o jogo

League of Legends é um jogo eletrônico do tipo MOBA (*Multiplayer Online Battle Arena*) desenvolvido e publicado pela Riot Games. Lançado em 2009, o game consiste em partidas competitivas compostas por dois times adversários de cinco jogadores cada.

O objetivo principal de cada equipe é atacar e destruir a torre principal da base adversária, chamada Nexus. Para isso o game é formado por "campeões", personagens jogáveis e que possuem características, habilidades e funções distintas. Cabe à equipe decidir qual a melhor combinação de campeões a fim de maximizar as chances de vitória. Além da torre principal, ficam espalhadas pelo território de cada time torres secundárias que protegem a base e atacam inimigos próximos. Também estão presentes *bots* aliados (Figura 4) e *mobs* neutros que oferecem recompensas estratégicas importantes.

Figura 4 - Dinâmica de ataque à torre com destaque para torre (1), campeões (2) e tropas (3).

Fonte: Autoria própria.

O desempenho de cada player durante uma partida pode ser mensurável e analisado por métricas disponíveis pelo próprio game. Dentre elas, podemos destacar:

- Kills (K): Quantidade de abates obtidos.
- Deaths (D): Número de vezes em que o player foi abatido.
- Assists (A): Quantidade de vezes em que o player ajudou um aliado a abater um inimigo.
- Kill to Death ratio (KD): Esse atributo mede a relação entre kills e deaths do player. Para valores maiores que 1, é considerado um bom desempenho. Para valores abaixo disso, é considerado um mal desempenho. É calculado por:

$$KD = \frac{K}{D} \tag{1}$$

- Kill-Death-Assist ratio (KDA): Essa métrica faz uma relação entre *kills*, *deaths* e *assists*, mostrando um panorama mais geral do player, sendo utilizado mais frequentemente que o KD. Pode ser calculado por:

$$KDA = \frac{K + A}{D} \tag{2}$$

- Dominance Factor (DF): O fator de dominância mede de forma somatória o impacto do player na partida. O cálculo dele penaliza mais uma morte do que favorece uma kill, assim é feito da seguinte forma:

$$DF = 2 \cdot K + A - 3 \cdot D \tag{3}$$

- Dominance Ratio (DR): A proporção de dominância faz uma razão entre os eventos de impacto positivo (kills e assists) e os abates. Dessa forma, é calculado por:

$$DR = \frac{2 \cdot K + A}{3 \cdot D} \tag{4}$$

3.3. Setup

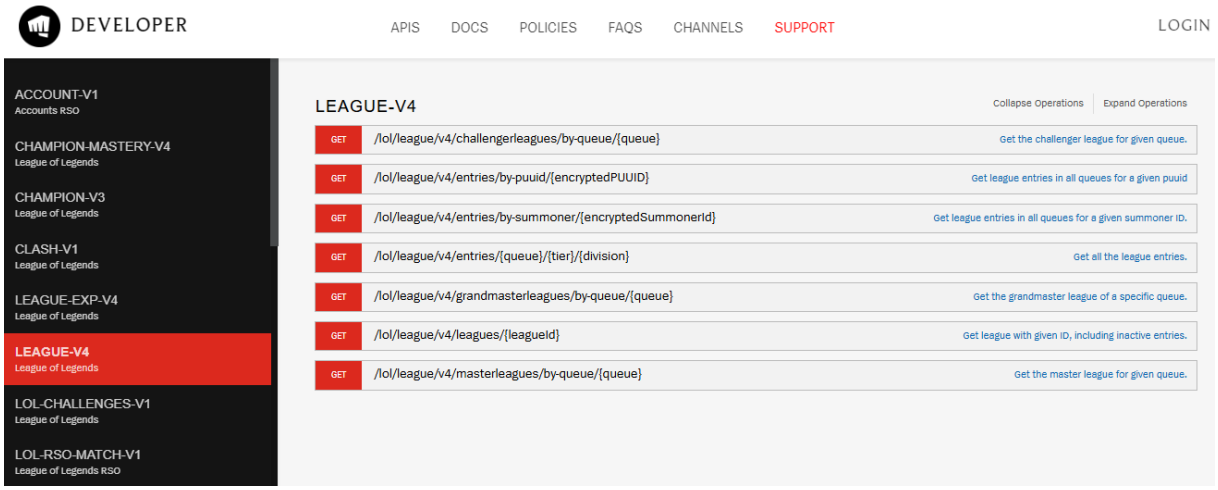
O setup utilizado para o desenvolvimento do projeto consistiu em um único hardware, sendo ele um notebook Acer Aspire 5, modelo A515-52G, com um processador Intel Core i5 de 8ª Geração, 12GB de memória RAM DDR3 e com uma placa gráfica (GPU) dedicada de 2GB.

O ambiente de desenvolvimento utilizado foi o Google Colaboratory (Colab). Ele é uma ferramenta gratuita do Google onde é possível programar diretamente no navegador, sem a necessidade de instalar o software localmente. Uma das praticidades dele é a possibilidade de compilar diferentes partes de um mesmo código isoladamente. Também possui a facilidade de se usar bibliotecas e de treinar modelos de Inteligência Artificial rodando em nuvem (Google, 2025). O código em Python utilizado pode ser encontrado pelo link no Apêndice A.

3.4. Coleta de dados

Assim como dito na seção anterior, o League of Legends é um dos jogos mais famosos da atualidade. Dessa forma, a própria desenvolvedora do game, a Riot Games (Figura 5), dispõe um portal para desenvolvedores, no qual são disponibilizadas diversas API's nativas para que os próprios players e entusiastas de jogos coletem dados para as mais diversas aplicações (Riot Games, 2025).

Figura 5 - Página inicial do Riot Developer Portal.



Fonte Riot Games, 2025

As API's disponíveis permitem a coleta dos mais diversos dados, desde o League of Legends até outros jogos da desenvolvedora. Em muitas delas, para fazer uma requisição é necessário passar algumas chaves como parâmetro. Assim, uma das primeiras etapas era definir quais seriam as mais relevantes para o projeto.

Ao fazer uma melhor avaliação de cada uma, foi possível notar que apenas com a Riot ID do jogador era possível coletar dados das N últimas partidas. Dentre as informações coletadas, além do ID passado como parâmetro, a requisição também retornava as estatísticas e todos os outros nove players que compõem a partida

3.4.1 Escolha dos players referência

Para este estudo, foram selecionados quatro jogadores profissionais de referência com base em critérios de representatividade no cenário competitivo brasileiro e viabilidade de coleta de dados. O primeiro escolhido foi *Robo*, um dos jogadores ainda ativos mais conhecidos do país e integrante da equipe LOUD, uma das principais organizações de *e-sports* do cenário nacional.

O segundo jogador selecionado foi *tinowns*, também da equipe LOUD, sendo o único outro jogador brasileiro da mesma equipe para o qual foi possível obter registros completos. Embora os times profissionais não sejam necessariamente compostos por jogadores da mesma nacionalidade, a limitação na coleta de dados de outros integrantes da equipe restringiu as opções disponíveis.

O terceiro jogador escolhido, dyNquedo, um integrante da equipe paiN Gaming - outra das maiores equipes brasileiras -, foi selecionado com base em sua posição coincidente com um dos dois jogadores anteriores, com o objetivo de permitir comparações diretas por função. Por fim, o quarto jogador, TitaN, foi escolhido de modo a representar uma terceira posição distinta, ampliando assim a diversidade de papéis analisados nas partidas.

Quadro 1 - Jogadores profissionais selecionados como referência

Jogador	Riot ID Atual	Posição	Time
Robo	ROBOCA DE SACOLA #BR1	Topo	LOUD
tinowns	tinowns #BR2	Meio	LOUD
dyNquedo	dyNNKaz #1010	Meio	PAIN
TitaN	paiN TitaN 10 #xsqdl	Atirador	PAIN

Fonte: Autoria própria.

3.5. Análise estatística

A análise estatística foi conduzida com o objetivo de identificar padrões, tendências e possíveis correlações entre as variáveis coletadas nas partidas. Inicialmente, aplicaram-se técnicas de estatística descritiva e exploratória, a fim de compreender o comportamento dos dados e orientar as etapas subsequentes do estudo.

Diversas ferramentas são amplamente utilizadas no campo da análise de dados e da ciência de dados, como Excel, SQL, Python e R. Neste trabalho, optou-se pela linguagem Python, em conjunto com o ambiente de desenvolvimento do Google Colab, por sua praticidade e integração com bibliotecas especializadas. Essa escolha permitiu a execução de todo o processo de ETL (Extração, Transformação e Carga de Dados) no mesmo ambiente em que também foi realizada a etapa de Machine Learning. Para isso, foram empregadas bibliotecas específicas para cada fase do desenvolvimento, facilitando a organização e a automação do fluxo de análise (ver Apêndice A para a lista completa de bibliotecas utilizadas).

Como etapa inicial da exploração dos dados, foram utilizadas ferramentas básicas de análise estatística, como histogramas e *boxplots*, para visualizar a distribuição das variáveis, além de medidas descritivas obtidas por meio do método "*describe*()", como média, mediana, desvio-padrão, máximo e mínimo. Complementarmente, foram analisadas a assimetria e a curtose das distribuições, permitindo avaliar a simetria e o grau de concentração dos dados em relação à média.

Para investigar possíveis relações entre variáveis numéricas, aplicou-se a correlação de Pearson, que mede o grau de associação linear entre pares de variáveis (Field, 2024). No caso de variáveis categóricas, foram utilizados o teste do qui-quadrado (χ^2), que avalia a independência entre variáveis (p-valor) assim como mostra Lee (2016), e o índice V de Cramer, que quantifica a força da associação entre categorias (Lee, 2016), como mostra o Quadro 2. Esses testes complementam a análise descritiva, fornecendo evidências estatísticas sobre a presença (ou ausência) de relações significativas nos dados coletados.

Quadro 2 - Interpretação dos valores do p-valor (χ^2) e do V de Cramér.

p-valor (χ^2)	**Significância Estatística**	**V de Cramér**	**Força da Associação**	**Interpretação Geral**
> 0,05	Não significativa	—	—	Sem associação estatisticamente significativa
≤ 0,05	Significativa	0,00 a 0,10	Muito fraca ou inexistente	Associação estatisticamente significativa, mas muito fraca
≤ 0,05	Significativa	0,10 a 0,30	Fraca	Associação estatisticamente significativa e fraca
≤ 0,05	Significativa	0,30 a 0,50	Moderada	Associação estatisticamente significativa e moderada
≤ 0,05	Significativa	0,50 a 0,70	Forte	Associação estatisticamente significativa e forte
≤ 0,05	Significativa	0,70 a 1,00	Muito forte	Associação estatisticamente significativa e muito forte

Fonte: Adaptado de Lee (2016)

3.6. Análise com Machine Learning

A técnica de Random Forest foi aplicada ao *dataset* com o objetivo de realizar previsões a partir das variáveis disponíveis, tanto em problemas de regressão quanto de classificação. Trata-se de um algoritmo baseado em *ensemble* de árvores de decisão, que combina múltiplos modelos para melhorar a precisão e reduzir o risco de *overfitting,* um problema comum em modelos de aprendizado de máquina que ocorre quando o modelo aprende demais os detalhes do conjunto de treino, incluindo ruídos e padrões específicos que não se generalizam para novos dados (Gerón, 2019).

No processo de avaliação, utilizou-se o método de validação cruzada K-Fold, juntamente com as métricas MAE (Erro Absoluto Médio) e RMSE (Raiz do Erro Quadrático Médio) para os modelos de regressão, e acurácia, precisão, recall e F1-Score nos casos de classificação. Depois de coletados os resultados utilizando o algoritmo Random Forest, o mesmo Dataset foi utilizado para treinar e coletar os resultados com os algoritmos de XGBoost Regressor e SVR para valores contínuos (como Kills, Deaths etc.). Em sequência o XGBoost Classifier e o SVG para classificar a variável de Team Win.

4. DATASET

O *dataset* final foi construído a partir da combinação dos quatro conjuntos de dados referentes aos jogadores profissionais de referência. Como mencionado anteriormente, cada partida coletada incluía também os dados dos outros nove jogadores que compunham os dois times.

Para cada linha registrada, foram extraídas as seguintes variáveis: Game ID, Duration, Kills, Deaths, Assists, Team ID, Team FK (First Kill) e Team Win. Em seguida, foram calculadas as métricas derivadas KD, KDA, DF e DR para cada jogador, com base nas estatísticas básicas. Com o objetivo de facilitar a aplicação dos modelos de classificação, as variáveis Team FK e Team Win foram convertidas de valores booleanos (True/False ou Win/Lose) para valores binários (1 ou 0).

4.1. Resultados da análise estatística

A análise estatística teve como objetivo descrever e visualizar o comportamento das variáveis coletadas, identificar padrões de desempenho e obter observações significativas. Nessa etapa, foram calculadas estatísticas descritivas como média, mediana, desvio-padrão, assimetria e curtose, de forma a caracterizar a distribuição dos dados e fornecer uma visão geral dos padrões presentes no conjunto analisado.

Durante o processo de análise das informações, o *dataset* foi tratado e reorganizado com diferentes filtros, visando um melhor agrupamento dos dados e, consequentemente, facilitando a visualização e interpretação dos resultados. Essa etapa incluiu, por exemplo, a segmentação dos dados por partida, por time e por resultado (vitória ou derrota), além da geração de métricas agregadas que permitiram uma análise mais estruturada dos padrões de desempenho observados nas partidas.

4.1.1 Análise com todos os players

Nessa primeira análise foi considerado o *dataset* original, contendo todas as informações de todos os *players*, de todas as partidas coletadas. A Tabela 1 mostra a saída gerada com informações resumidas de estatísticas já calculadas.

Tabela 1 - Variáveis estatísticas considerando todos os players (continua)

Estatística	Duração	Kills	Deaths	Assists	KD	KDA	DF	DR
count	800	800	800	800	800	800	800	800

Tabela 1 - Variáveis estatísticas considerando todos os players (conclusão)

mean	1597,988	5,68	5,6988	9,1625	1,446	3,6723	9,0913	2,0381
std	354,6812	4,41	3,1509	7,1467	1,7619	3,686	16,369	3,1335
min	149	0	0	0	0	0	-29	0
25%	1390,25	2	4	4	0,4	1,4286	-3	0,6667
50%	1631	5	5	7,5	0,875	2,5147	8	1,6667
75%	1822,75	8	8	12	1,8616	4,25	20	2,0556
max	2328	28	18	54	13	32	68	31

Fonte: Autoria própria.

É importante destacar que a média de Kills e Deaths por jogador apresenta valores próximos, não por refletirem um equilíbrio de desempenho, mas por serem estatisticamente relacionadas. Cada *kill* de um jogador representa, obrigatoriamente, a morte de um oponente. No entanto, a recíproca não é sempre verdadeira, já que um jogador pode, ainda que em eventos raros, morrer por ações de *bots* ou *mobs* neutros, o que não é computado como uma *kill* de outro jogador. Dessa forma, a proximidade entre os valores médios de Kills e Deaths indica que a presença de tropas controladas por *bots* ou *mobs* neutros não tem impacto significativo nas estatísticas de abates entre jogadores. Por outro lado, a eliminação de tropas inimigas, como *minions* e monstros da selva, contribui diretamente para o acúmulo de ouro pela equipe, o que pode influenciar de maneira decisiva o desenrolar da partida, ao permitir a compra de itens e o fortalecimento dos campeões.

A métrica KDA, com média de 3,67 e mediana de 2,51, demonstra uma distribuição assimétrica, com presença de jogadores com performances bastante superiores à média (valores máximos de KDA de até 32). As métricas derivadas DF e DR apresentaram valores mínimos negativos ou extremamente baixos (DF chegando a -29), o que reforça a presença de outliers e de desempenhos muito abaixo da média. O valor máximo de DR (31) e a curtose esperada dessas variáveis indicam caudas longas e concentração de casos extremos, especialmente quando o número de mortes é baixo.

Em sequência, o *dataset* foi agrupado por times e por partida, de forma que os valores de Kills, Deaths e Assists fossem a soma de todos os players do time, como mostra a Tabela 2. Assim, o *dataset* adquiriu um formato em que uma única partida (Game ID) fosse representada por duas linhas, sendo uma de cada time.

Tabela 2 - Variáveis estatísticas por time/por partida

Estatística	Duração	Kills	Deaths	Assists
count	160	160	160	160
mean	1597,9875	28,375	28,49375	45,8125
std	355,57234	12,25553	12,26187	27,74325
min	149	0	0	0
25%	1390,25	21	21	27,75
50%	1631	28	28	41,5
75%	1822,75	36,25	36,25	59
max	2328	75	75	154

Fonte: Autoria própria.

A análise dessa tabela isoladamente não traz pontos significativamente relevantes. Contudo ela foi usada como comparativo para os *datasets* filtrados por times vencedores e times perdedores, assim como mostram as Tabelas 3 e 4, respectivamente.

Tabela 3 - Variáveis estatísticas times vencedores

Estatística	Duração	Kills	Deaths	Assists
count	80	80	80	80
mean	1597,9875	34,1875	22,6875	55,9
std	356,69579	10,31049	11,29982	26,78399
min	149	1	0	1
0,25	1390,25	27,75	15	36,75
0,5	1631	33,5	22,5	53,5
0,75	1822,75	40	28	65,25
max	2328	75	69	154

Fonte: Autoria própria.

Tabela 4 - Variáveis estatísticas times perdedores

Estatística	Duração	Kills	Deaths	Assists
count	80	80	80	80
mean	1597,9875	22,5625	34,3	35,725
std	356,69579	11,2945	10,32681	25,0129
min	149	0	1	0
25%	1390,25	15	27,75	19,75
50%	1631	22,5	34	33,5
75%	1822,75	28	40	45
max	2328	69	75	151

Fonte: Autoria própria.

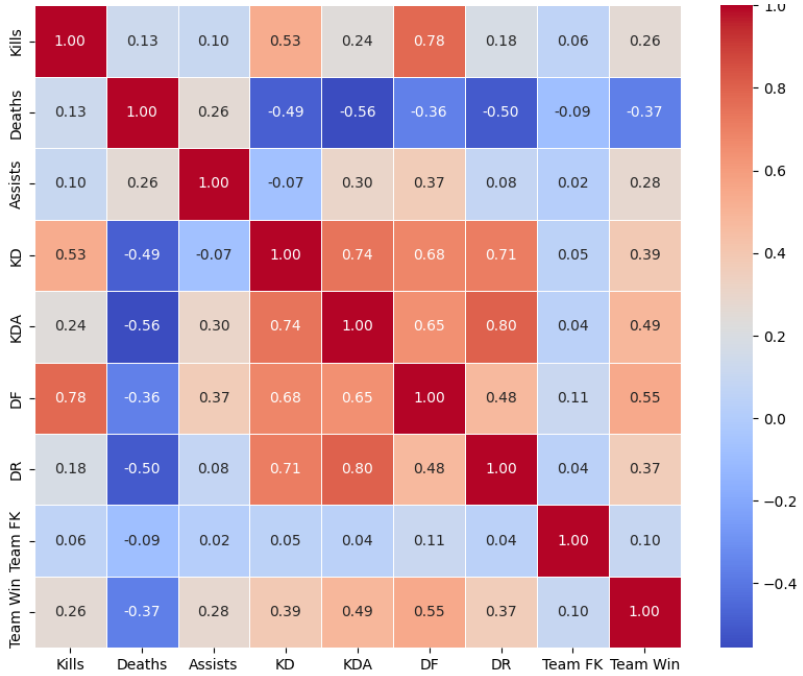
Quando os dados são segmentados por times vencedores e perdedores, observa-se uma diferença já antecipada nas variáveis Kills e Deaths. Os times vencedores registraram, em

média, 34,19 kills, enquanto os perdedores apresentaram 22,56 — uma diferença proporcional de aproximadamente 66% a favor dos vencedores. Essa mesma lógica se aplica à variável Deaths, invertendo os valores entre os grupos.

A variável Assists, por sua vez, apresentou a maior diferença em termos absolutos: 55,9 nos times vencedores contra 35,73 nos perdedores — cerca de 20 assistências a mais por partida. Apesar de ser uma métrica distinta, a proporcionalidade relativa entre os grupos se manteve semelhante à observada em Kills e Deaths: os vencedores superaram os perdedores em cerca de 20% em relação à média geral da variável.

Esse alinhamento proporcional sugere que, embora Assists não sejam estatisticamente dependentes de Kills ou Deaths, o desempenho coletivo — representado pelas assistências — acompanha a superioridade ofensiva dos times vencedores, reforçando seu potencial como variável explicativa no contexto das partidas. Com essas observações destacadas, foi feita uma matriz de correlação, utilizando o coeficiente de correlação de Pearson. Essa métrica mede a relação linear entre duas variáveis, resultando numa saída com valores entre -1 e 1, onde quanto mais próximo de 1 maior é a relação direta entre elas. Do contrário, quando mais próximo de -1 mais inversamente relacionadas. A Figura 6 mostra a relação de todas as variáveis relacionadas entre si, ainda para o *dataset* considerando todos os players.

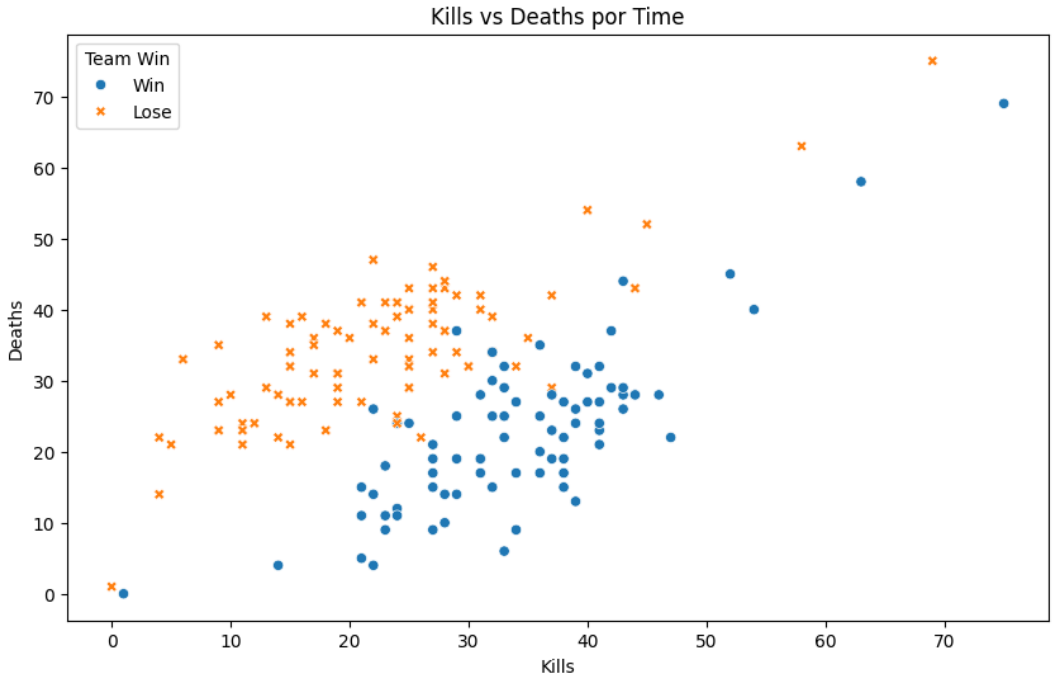
Figura 6 - Matriz de Correlação das Variáveis utilizando todos os Jogadores.



Fonte: Autoria própria

Assim como esperado, as variáveis dependentes, como KD, KDA, DR e DF, apresentam forte correlação (direta ou inversa) com as variáveis independentes Kills, Deaths e Assists. Observa-se também uma relação relevante da variável Team Win com essas métricas. De maneira geral, uma Death influencia negativamente de forma mais impactatnte do que uma Kill influencia positivamente, especialmente na análise individual. Isso é evidenciado pela fórmula de DF, que penaliza diretamente as mortes. Contudo, quando os dados são observados no espectro coletivo, ou seja, somando os valores totais de Kills e Deaths de cada time, percebe-se um padrão claro: times vencedores tendem a manter um volume de Kills superior ao de Deaths, como evidencia a Figura 7. Portanto, enquanto no nível individual o foco em reduzir mortes é mais determinante, no contexto coletivo, maximizar o saldo positivo entre Kills e Deaths continua sendo crucial para a vitória.

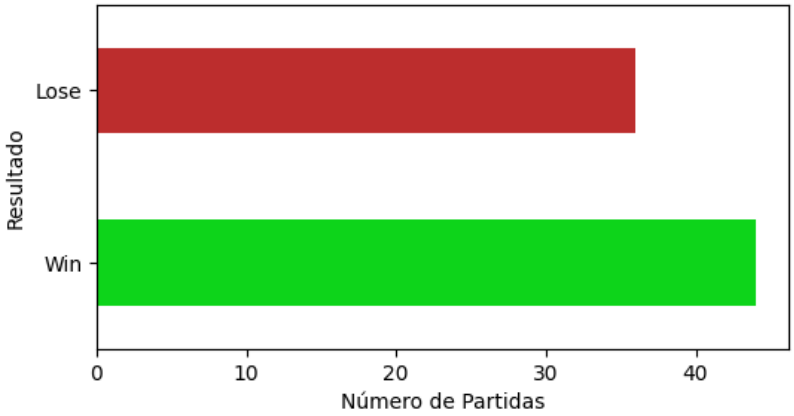
Figura 7 - Relação entre Kills e Deaths por Time segmentada por resultado



Fonte: Autoria própria.

Outra observação a se fazer é a ausência de correlação em First Kill e as demais variáveis, onde a maior (DF) possui um valor de 0,11 apenas. Visto isso, o *dataset* foi filtrado para permitir a visualização (como mostra na Figura 8) da relação entre os times que obtiveram a primeira *kill*, se venceram a partida ou não.

Figura 8 - Análise de dependência entre First Kill e vitória na partida (Team Win)



Fonte: Autoria própria.

Essa análise resultou em 44 vitórias e 36 derrotas para os times que obtiveram a First Kill. Como se trata de duas variáveis categóricas e binárias, foi aplicado um teste de hipótese com o objetivo de verificar se há associação estatística significativa entre elas. Para isso, utilizou-se o teste do Qui-quadrado, que permite identificar a existência de dependência entre as categorias. Além disso, foi calculado o V de Cramer, a fim de mensurar a intensidade da relação observada.

Com isso, foram obtidos os valores de 0,6284 e 0,0875 para o Qui2 e V de Cramer, respectivamente. O valor 0,6284, assim como mostrado nos tópicos anteriores desse estudo, mostra uma baixa associação entre as variáveis (>0,05), mostrando que a probabilidade de possuírem valores não relacionados é maior do que serem correspondentes e proporcionais. E o valor de Cramer mostrando uma fraca relação (<0,1) entre elas confirma esse levantamento. Contudo, é válido ressaltar que, para um número baixo de amostras (80 partidas) essas métricas podem não ser tão precisas, visto que elas são expressivamente sensíveis a pequenas variações.

4.1.2 Análise dos quatro jogadores profissionais

A fim de complementar as análises de variáveis estatísticas, os *datasets* foram filtrados para se avaliar isoladamente as métricas de cada um dos quatro jogadores profissionais usados para coletar os dados. As tabelas e gráficos coletados para os quatro jogadores encontram-se no Apêndice B.

Primeiramente, os gráficos de *Robo* indicam uma distribuição assimétrica nas métricas compostas (KD, KDA e DR), com a presença de outliers positivos. A baixa variabilidade em Deaths e a maior dispersão em Kills e Assists sugerem um perfil ofensivo com baixo risco.

Para o segundo player, ainda da equipe LOUD, *tinowns*, os gráficos apresentaram distribuições com menor variabilidade em Kills, mas com assistências mais distribuídas. A métrica DR também apresentou picos pontuais, indicando desempenho dominante em algumas partidas. As caudas superiores mais longas nos *boxplots* de KDA e DR reforçam essa característica.

Já da equipe Pain, o jogador *dyNquedo* apresentou distribuição mais equilibrada entre as métricas KDA e DR, com menor frequência de outliers. Os histogramas de Assists indicam contribuição regular em jogadas coletivas, enquanto os *boxplots* demonstram consistência em performance, com menos variações extremas. Com resultados parecidos do anterior, o terceiro *player* apresentou um desempenho mais assistencial, onde favorece o manejo e o controle da partida. Como visto no início do estudo, esses dois jogadores atuam na posição de Meio, o que colabora com esse perfil menos ofensivo e mais equilibrado.

Por último *TitaN* demonstrou ampla variabilidade em Kills, com a presença de outliers evidentes, característica de seu papel de carregador principal. Fazendo correlação com a sua posição de Atirador, na qual ele realiza kills que, apesar de menos frequentes, são decisivas e exercem maior impacto no resultado da partida do que um alto volume de abates com menor influência estratégica. As métricas KDA e DR apresentaram maior assimetria, com valores máximos mais distantes, sugerindo impacto direto nas partidas onde se destacou.

4.2. Resultados com Algoritmos de Machine Learning

Nesta seção, são apresentados os resultados obtidos a partir da aplicação de algoritmos de aprendizado de máquina utilizando o conjunto completo de dados coletados.

4.2.1 Random Forest

O algoritmo de Random Forest pôde ser implementado diretamente no ambiente de desenvolvimento utilizando as respectivas bibliotecas. Como dito anteriormente, foi utilizado o método de validação cruzada (K-fold Cross Validation).

O particionamento foi definido de forma a garantir que uma quantidade suficiente de dados fosse destinada tanto ao treinamento quanto à validação, considerando o tamanho da base de dados e buscando minimizar viés e variância nos modelos. Sendo assim, $k = 5$ foi o valor mais adequado para utilizar dentro do método, para dividir o conjunto de dados em cinco partes. Em cada iteração, quatro partes (80%) foram utilizadas para o treinamento do modelo e a parte

restante (20%) para validação do modelo, de forma alternada até que todos os subconjuntos fossem utilizados como teste uma vez

Assim, primeiramente foi utilizado o Random Forest Regressor para mensurar previsões de números contínuos com o *dataset* sem a remoção de *outlier*s. Dessa forma, as variáveis coletadas utilizadas no modelo foram Kills, Deaths, Assists, KD, KDA, DF e DR. A Tabela 5 mostra os valores obtidos em termos de MAE Médio e RMSE Médio.

Tabela 5 - Resultados obtidos para o Random Forest Regressor com *outliers*.

Variável	MAE Médio	RMSE Médio
Kills	0.326338	0.760682
Deaths	0.401062	0.710490
Assists	1.156300	2.017899
KD	0.101366	0.246072
KDA	0.216477	0.621040
DF	1.042037	2.025682
DR	**0.091940**	**0.358563**

Fonte: Autoria própria.

Os resultados obtidos com o Random Forest Regressor, mesmo mantendo os outliers no conjunto de dados, evidenciam boa capacidade preditiva em métricas como KD (MAE: 0,01) e DR (MAE: 0,09), indicando estabilidade e menor sensibilidade a variações extremas. Isso se assemelha aos resultados obtidos com a análise estatística, onde as variáveis compostas apresentam maior previsibilidade. Por outro lado, variáveis como Assists (RMSE: 2,01) e DF (RMSE: 2,03) apresentaram os maiores erros, o que sugere alta dispersão e provável influência de fatores coletivos ou ruídos nas partidas. Na análise anterior a média de Assists obedecia a proporcionalidade de Kills/Deaths, mas no algoritmo ela mostrou-se menos previsível.

Após a primeira rodada de testes, foi realizada uma nova abordagem com a remoção dos outliers do *dataset*, com o objetivo de avaliar seu impacto na performance do modelo. A Tabela 6 faz uma comparação entre os resultados do RF Regressor com e sem *outliers*.

Tabela 6 - Comparação dos resultados com e sem *outliers* para o Random Forest Regressor.

Variável	MAE (com)	MAE (sem)	Δ MAE	RMSE (com)	RMSE (sem)	Δ RMSE
Kills	0.326	0.340	+	0.760	0.800	+
Deaths	0.401	0.420	+	0.710	0.752	+
Assists	1.156	1.130	-	2.017	1.985	-
KD	0.101	0.096	-	0.246	0.224	-
KDA	0.216	0.224	+	0.621	0.732	+
DF	1.042	1.056	+	2.025	1.986	-
DR	**0.092**	**0.085**	**-**	**0.359**	**0.319**	**-**

Fonte: Autoria própria.

Após a aplicação do Random Forest Regressor sobre o *dataset* com remoção de *outliers*, observou-se, de modo geral, uma piora nos valores de MAE e RMSE para a maioria das variáveis. Em particular, as métricas Kills, Deaths e KDA apresentaram aumento nos erros médios, o que indica que os *outliers* previamente presentes contribuíram com informações relevantes para o modelo. Por outro lado, variáveis como KD, DR e Assists tiveram melhora ou estabilidade nos erros, sugerindo que são menos dependentes de extremos e mais consistentes em sua distribuição.

Após a utilização do RF Regressor, foi utilizado o modelo Classifier para rodar previsões sobre a variável de Team Win, utilizando métricas de Acurácia, Precision, Recall e F1-Score na validação do modelo. Na Tabela 7 é possível ver a comparação entre os modelos, com e sem *outliers*.

Tabela 7 - Comparação dos resultados com e sem *outliers* para o Random Forest Classifier medindo a variável Team Win.

Métrica	Com Outliers	Sem Outliers	Δ
Acurácia	0.7562	0.7525	–0.0037
Precision	**0.7517**	**0.7450**	**–0.0067**
Recall	0.7544	0.7639	+0.0095
F1-Score	0.7517	0.7521	+0.0004

Fonte: Autoria própria

Essa análise evidencia que, neste caso específico, a presença de *outliers* não comprometeu o modelo e, em algumas situações, contribuiu positivamente para sua performance preditiva. Com o objetivo de ampliar a análise comparativa entre algoritmos de aprendizado supervisionado, também foram aplicados os modelos XGBoost e SVM, utilizando os mesmos dados e métricas adotadas na avaliação anterior.

4.2.2 XGBoost

Como alternativa ao Random Forest, foi também aplicado o modelo XGBoost tanto para tarefas de regressão quanto para classificação. Os testes seguiram a mesma lógica anterior, com validação cruzada por K-Fold e uso do *dataset* sem a remoção de *outliers*. Nos outros modelos foram apenas consideradas as variáveis dependentes (Kills, Deaths e Assists), dada a alta previsibilidade das variáveis compostas, como mostra a Tabela 8.

Tabela 8 - XGBoost Regressor para variáveis contínuas.

Métricas	MAE Médio	RMSE Médio
Kills	**0.2814**	**0.6799**
Deaths	0.3408	0.5914
Assists	0.9452	1.7234

Fonte: Autoria própria.

Os resultados obtidos com o XGBoost Regressor indicam um bom desempenho, especialmente para as variáveis Kills (MAE: 0.2814) e Deaths (MAE: 0.3408), que apresentaram os menores erros médios em comparação aos modelos anteriores. A variável Assists seguiu apresentando maior dispersão (RMSE: 1.7234), mas ainda com desempenho superior ao observado no Random Forest. A Tabela 9 abaixo mostra os resultados obtidos para a previsão de vitória ou derrota utilizando o modo de classificação.

Tabela 9 - XGBoost Classifier para Team Win.

Métrica	Valor Médio
Acurácia	0.731250
Precision	**0.723839**
Recall	0.737528
F1-Score	0.729941

Fonte: Autoria própria.

No caso da classificação do resultado da partida (Team Win), o XGBoost alcançou uma acurácia de 0.7312 e F1-score de 0.7299, levemente inferiores aos valores obtidos com o Random Forest. Apesar disso, a métrica Recall atingiu seu maior valor (0.7375) entre todos os modelos testados, o que indica uma maior sensibilidade na identificação de vitórias. De maneira geral, o XGBoost demonstrou estabilidade mesmo na presença de outliers, com métricas consistentes e desempenho competitivo tanto na regressão quanto na classificação.

4.2.3 SVM

Por fim, foi testado também o algoritmo Support Vector Machine (SVM), utilizando tanto o SVR para tarefas de regressão quanto o SVC para classificação. Os testes utilizaram o mesmo conjunto de dados utilizado nos modelos anteriores, mantendo os *outliers* e aplicando validação cruzada com K-Fold. Os resultados encontram-se na Tabela 10 abaixo.

Tabela 10 - SVR para Kill, Death e Assists.

Variável	MAE Médio	RMSE Médio
Kills	0.303156	0.685157
Deaths	**0.242815**	**0.523741**
Assists	0.515532	1.216637

Fonte: Autoria própria.

Os testes com o SVR apresentaram erros baixos em todas as variáveis, com destaque para Deaths, que obteve os menores valores de MAE (0.2428) e RMSE (0.5237), superando os demais modelos nesse aspecto. Apesar de Assists continuar com maior dispersão, o modelo foi mais estável que os anteriores, mantendo bom desempenho mesmo com *outliers*.

Já na classificação com SVC, como mostra a Tabela 11, o desempenho foi o melhor entre todos os algoritmos testados. O modelo atingiu acurácia de 78,37%, além de valores elevados de Recall (0.8078) e F1-score (0.7865), indicando alta capacidade de identificar corretamente os times vencedores sem comprometer a precisão geral do modelo.

Tabela 11 - SVC para Team Win. (continua)

Métrica	Valor Médio
Acurácia	0.783750
Precision	**0.766894**

Tabela 11 - SVC para Team Win (conclusão)

Recall	0.807775
F1-Score	0.786474

Fonte: Autoria própria.

De forma geral, o SVM se mostrou o modelo um pouco mais eficaz, especialmente na tarefa de classificação, apresentando os melhores resultados entre os algoritmos avaliados.

5. CONSIDERAÇÕES FINAIS

Este trabalho teve como objetivo investigar o potencial da análise estatística e da inteligência artificial na geração de inferências a partir de dados públicos do jogo League of Legends. Ao longo do projeto, foram aplicadas técnicas de estatística descritiva, testes de hipótese e algoritmos de aprendizado de máquina para modelar comportamentos e prever variáveis relevantes em partidas competitivas.

A escolha do League of Legends como base do estudo se mostrou acertada, dado o alto volume de informações disponíveis e a estrutura das partidas, que favorece a coleta de dados padronizados e comparáveis entre diferentes contextos. A partir da API pública da Riot Games, foi possível extrair dados de desempenho de quatro jogadores profissionais brasileiros e, por consequência, de centenas de partidas envolvendo múltiplos perfis de jogadores.

Na etapa de análise estatística, foram observadas tendências importantes, como a forte associação entre métricas compostas (KD, KDA, DR) e variáveis básicas (Kills, Deaths, Assists), além da identificação de outliers significativos que marcaram desempenhos extremos. Uma das observações mais relevantes foi a alta diferença no número de assistências entre times vencedores e perdedores, sugerindo que a participação coletiva pode ser tão ou mais decisiva que o desempenho individual.

Com base nessas análises, foi possível aplicar os modelos de machine learning — Random Forest, XGBoost e SVM — tanto para regressão (previsão de variáveis numéricas) quanto para classificação (previsão da vitória ou derrota da equipe). Os modelos foram avaliados com validação cruzada e métricas apropriadas para cada tarefa, como MAE e RMSE para regressão, e acurácia, precisão, recall e F1-score para classificação. Nas tabelas 12 e 13 é possível ver as comparações entre os modelos para regressão e classificação, respectivamente.

Tabela 12 - Comparativo entre algoritmos para regressão (Random Forest, XGBoost, SVM).

	Random Forest		**XGBoost**		**SVM**	
Variável	**MAE Médio**	**RMSE Médio**	**MAE Médio**	**RMSE Médio**	**MAE Médio**	**RMSE Médio**
Kills	0.326338	0.760682	0.2814	0.6799	0.303156	0.685157
Deaths	0.401062	0.710490	0.3408	0.5914	0.242815	0.523741
Assists	1.156300	2.017899	0.9452	17.234	0.515532	1.216637

Fonte: Autoria própria

Tabela 13 - Comparativo entre algoritmos para classificação de Team Win.

	Random Forest	**XGBoost**	**SVM**
Acurácia	0.7562	0.731250	0.783750
Precision	0.7517	0.723839	0.766894
Recall	0.7544	0.737528	0.807775
F1-Score	0.7517	0.729941	0.786474

Fonte: Autoria própria

Dentre os modelos testados, o SVM (SVC) apresentou os melhores resultados para classificação, atingindo acurácia de 78,4% e recall de 80,8%, o que indica alta capacidade de prever corretamente os casos de vitória. Já nas tarefas de regressão, o XGBoost teve o melhor desempenho geral em Kills e Deaths, enquanto o SVR se destacou para a variável Deaths com o menor MAE entre todos os testes. Outro ponto relevante foi o impacto dos outliers: ao contrário do esperado, a remoção desses valores extremos piorou o desempenho dos modelos em várias métricas, o que demonstra que, neste contexto, os outliers carregavam informação útil — especialmente para variáveis compostas como DF e KDA.

Na análise por jogador, foi possível observar padrões condizentes com o estilo de jogo de cada um. Jogadores ofensivos, como *TitaN* e *Robo*, apresentaram maior variabilidade em Kills, enquanto jogadores de meio, como *dyNquedo* e *tinowns*, tiveram desempenho mais equilibrado, com assistências mais distribuídas e menor incidência de outliers. Entre as principais limitações do estudo, destaca-se o número reduzido de jogadores analisados e a dependência de uma API externa com restrições de acesso. Além disso, a ausência de variáveis contextuais, como composição de equipe ou ranking do jogador, limita a profundidade das inferências.

Como trabalhos futuros, sugere-se:

- A ampliação da base de dados, incluindo jogadores casuais e de outros rankings;
- A aplicação de técnicas não supervisionadas, como clusterização de perfis;
- O uso de redes neurais e algoritmos mais complexos;
- Criação de um sistema automatizado de predição e visualização para uso em tempo real.

Em conclusão, os resultados demonstraram que é viável e eficaz utilizar IA e estatística para extrair insights relevantes de partidas de League of Legends, confirmando o potencial

dessa abordagem para auxiliar em análises estratégicas, treinamentos e até mesmo no desenvolvimento de ferramentas de suporte para jogadores e equipes.

REFERÊNCIAS

CHAMMA, W. D. S.; BATISTELLA, D.; CRISIGIOVANNI, E. L.; VICTORINO, H. S.; LIMA, V. A. **Aprendizado de máquina aplicado em imagens de satélite para classificação de telhados**. Brazilian Journal of Development, Curitiba, v. 7, n. 7, p. 72558–72576, jul. 2021. DOI: 10.34117/bjdv7n7-437.

DATACAMP. **A comprehensive guide to K-Fold Cross Validation**. 2024. Disponível em: https://www.datacamp.com/tutorial/k-fold-cross-validation.

DATA INSIGHTS**. Pesquisa Game Brasil 2022**. São Paulo: Go Gamers, 2022. Disponível em: https://www.pesquisagamebrasil.com.br/

DURACHMAN, Y.; RAHMAN, A. W. A. **Investigating the Impact of Gameplay Hours on Player Recommendations in Steam Games: A Comparative Analysis Using Logistic Regression and Random Forest Classifiers**. International Journal of Research in Management, v. 5, n. 1, p. 15–28, 2023. Disponível em: https://ijrm.net/index.php/ijrm/article/view/21/20.

FIELD, A. **Discovering Statistics Using IBM SPSS Statistics**. 6. ed. London: SAGE Publications, 2024.

GÉRON, A. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems**. 2. ed. Sebastopol: O'Reilly Media, 2019.

GOOGLE. **Google Colaboratory** (2025). Disponível em: https://colab.research.google.com/

IBM. **What is XGBoost?** 2024. Disponível em: https://www.ibm.com/think/topics/xgboost.

LEE, D. K. **Alternatives to P value: confidence interval and effect size**. Korean Journal of Anesthesiology, v. 69, n. 6, p. 555–562, 2016. Disponível em: https://doi.org/10.4097/kjae.2016.69.6.555

LI, Z.; WANG, D.; ZOU, Y. **Game Classification and Analysis Based on Machine Learning-Based Methods**. In: INTERNATIONAL CONFERENCE ON DATA SCIENCE AND ENGINEERING (ICDSE 2024)

NEWZOO. **Global Games Market Report: The VR & Metaverse Edition – Free Version**. Amsterdam: Newzoo, 2021. Disponível em: https://newzoo.com/products/reports/global-games-market-report/.

QIU, Y.; GONG, Y.; LIU, G. **Behavior Analysis of Mobile Game Users: A Temporal and Structural Perspective**. arXiv preprint, 2024. Disponível em: https://arxiv.org/abs/2407.11772.

RÍO, A. F. del; CHEN, P. P.; PERIÁÑEZ, Á. **Predicting Player Engagement in Video Games with Survival Analysis and Deep Learning**. arXiv preprint, 2019. Disponível em: https://arxiv.org/abs/1907.03870.

RIOT GAMES**. Riot Developer Portal: APIs.** 2025. Disponível em: https://developer.riotgames.com/apis.

SAPIENZA, A.; BESSI, A.; FERRARA, E. **Non-negative Tensor Factorization for Human Behavioral Pattern Mining in Online Games**. arXiv preprint, 2017. Disponível em: https://arxiv.org/abs/1702.05695.

SEMENOV, A.; ROMOV, P.; KOROLEV, S.; YASHKOV, D.; NEKLYUDOV, K. **Performance of Machine Learning Algorithms in Predicting Game Outcome from Drafts in Dota 2**. Procedia Computer Science, v. 101, p. 250–257, 2016.

TWENTY ONE PILOTS. Stressed Out. In: Blurryface [CD]. Los Angeles: Fueled by Ramen, 2015. Faixa 2.

UOL. Com 334 **milhões de espectadores, audiência do Mundial de "LOL" superou NBA**, dez. 2015. Disponível em: https://www.uol.com.br/start/ultimas-noticias/2015/12/10/final-de-mundial-de-league-of-legends-teve-mais-espectadores-do-que-nba.htm

XPETU. **Win probability estimation for strategic decision-making in esports**. Master's Thesis (Mathematics and Operations Research) – Aalto University, School of Science, 2024.

YANG, P.; HARRISON, B.; ROBERTS, D. L. **Identifying patterns in combat that are predictive of success in MOBA games**. 2016 IEEE Conference on Computational Intelligence and Games (CIG).

ZHU, L.; ZHOU, X.; ZHANG, C. **Rapid identification of high-quality marine shale gas reservoirs based on the oversampling method and random forest algorithm**. Artificial Intelligence in Geosciences, v. 2, p. 76–81, 2021.

APÊNDICE A – Código e bibliotecas utilizados

O código em Python utilizado nesse estudo encontra-se no repositório do GitHub. Disponível em: https://github.com/rafinha-es/PFC_Game_Data_Science

Quadro A1 – Lista de bibliotecas e módulos utilizados

Nome	Descrição
requests	Biblioteca externa utilizada para realizar requisições HTTP.
pandas	Biblioteca externa para manipulação e análise de dados em formato tabular (DataFrames).
csv	Módulo da biblioteca padrão do Python usado para leitura e escrita de arquivos .csv.
numpy	Biblioteca externa usada para operações matemáticas e vetoriais com arrays numéricos.
os	Módulo da biblioteca padrão do Python que permite interações com o sistema operacional.
seaborn	Biblioteca externa baseada em Matplotlib, voltada à visualização estatística de dados.
matplotlib.pyplot	Módulo da biblioteca matplotlib usado como base para criação de gráficos e visualizações.
scipy.stats	Módulo da biblioteca scipy utilizado para testes estatísticos, como o teste do qui-quadrado.
sklearn.ensemble	Módulo da biblioteca scikit-learn que implementa algoritmos ensemble como Random Forest.
sklearn.model_selection	Módulo do scikit-learn utilizado para técnicas de validação como o K-Fold.
sklearn.metrics	Módulo do scikit-learn usado para avaliar o desempenho de modelos (acurácia, F1, MAE, etc).
xgboost	Biblioteca externa especializada em aprendizado de máquina baseada em árvores (XGBoost).
sklearn.svm	Módulo do scikit-learn que implementa algoritmos SVM para regressão e classificação.
sklearn.preprocessing	Módulo do scikit-learn que fornece ferramentas de normalização e padronização de dados.

APÊNDICE B – Gráficos coletados das variáveis de cada jogador profissional

Figura B1 – Histograma para as variáveis Kill, Death e Assists do jogador *Robo*

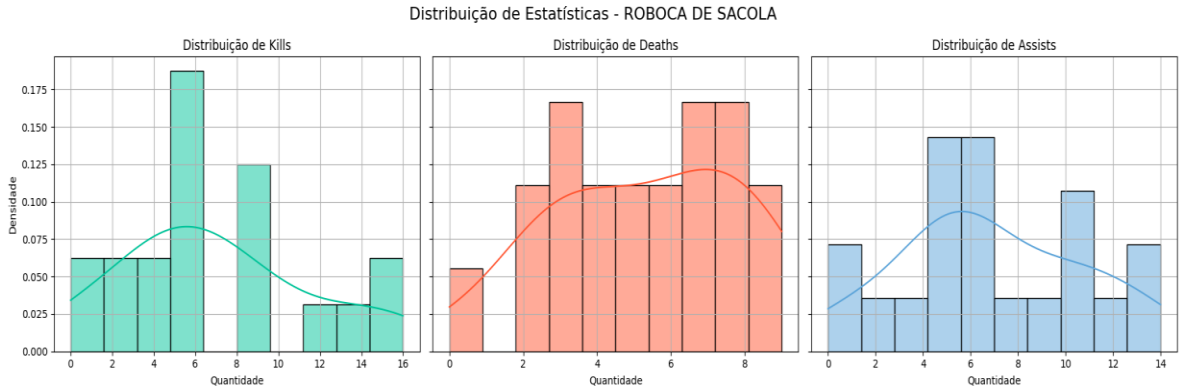


Figura B2 – Histograma para as variáveis KD, KDA e DR do jogador *Robo*

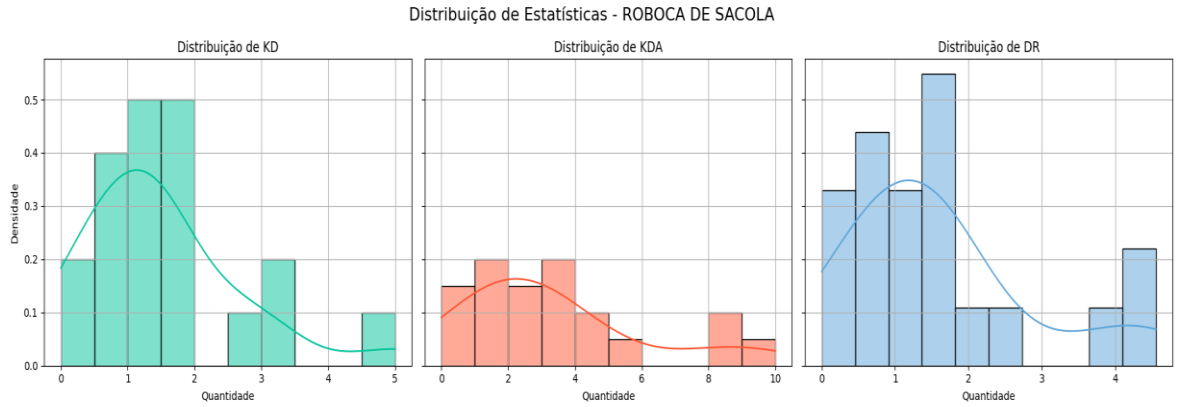


Figura B3 – Histograma para a variável DF do jogador *Robo*

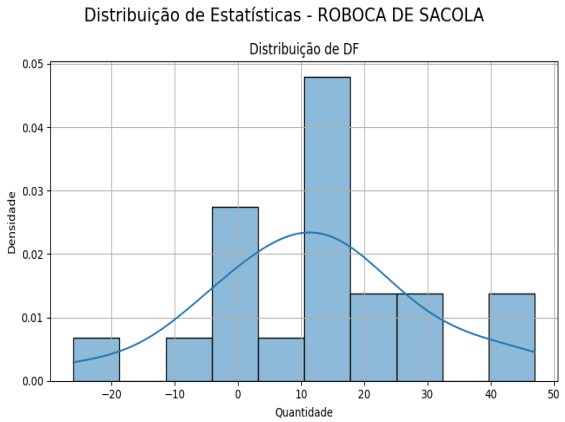


Figura B4 – Boxplot para as variáveis Kill, Death e Assists do jogador *Robo*

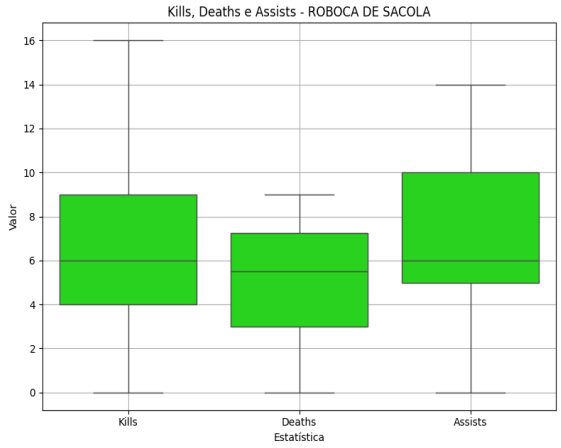


Figura B5 – Boxplot para as variáveis KD, KDA e DR do jogador *Robo*

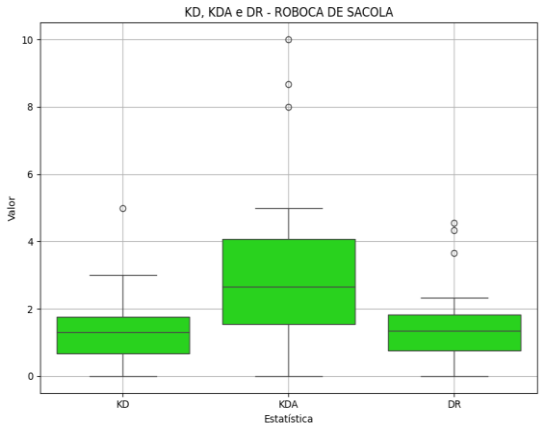


Figura B6 – Boxplot para a variável DF do jogador *Robo*

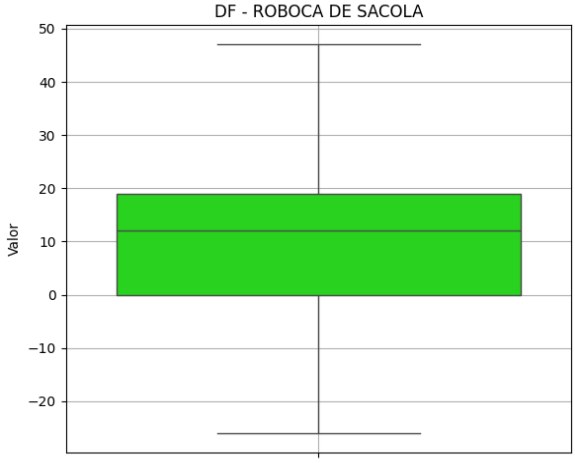


Figura B7 – Histograma para as variáveis Kill, Death e Assists do jogador *tinowns*

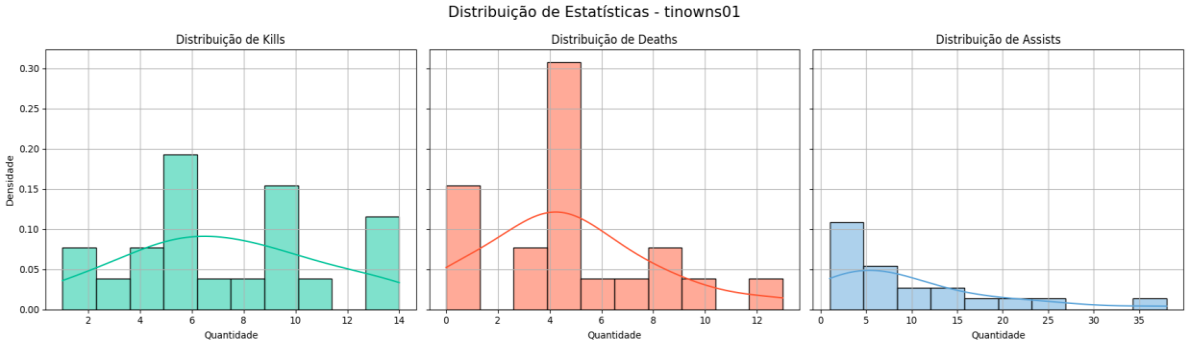


Figura B8 – Histograma para as variáveis KD, KDA e DR do jogador *tinowns*

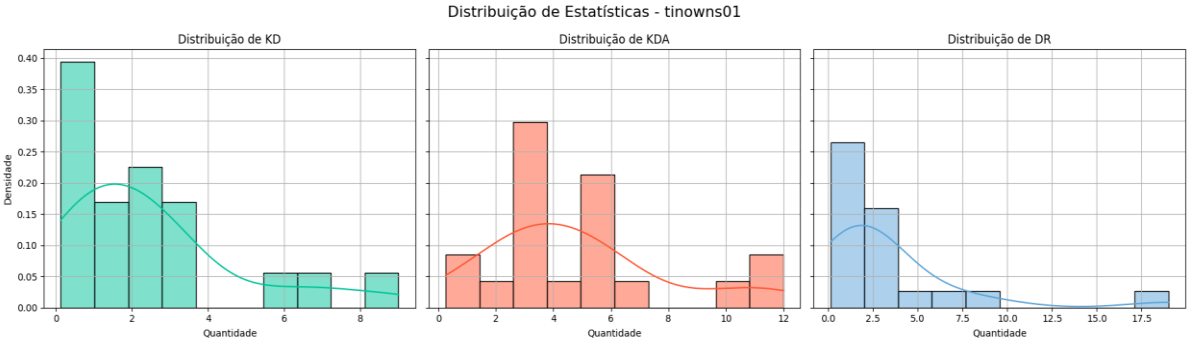


Figura B9 – Histograma para a variável DF do jogador *tinowns*

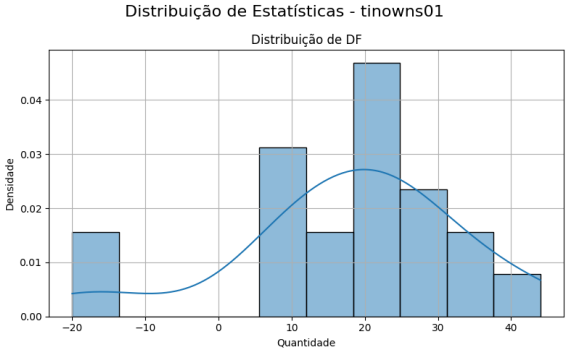


Figura B10 – Boxplot para as variáveis Kill, Death e Assists do jogador *tinowns*

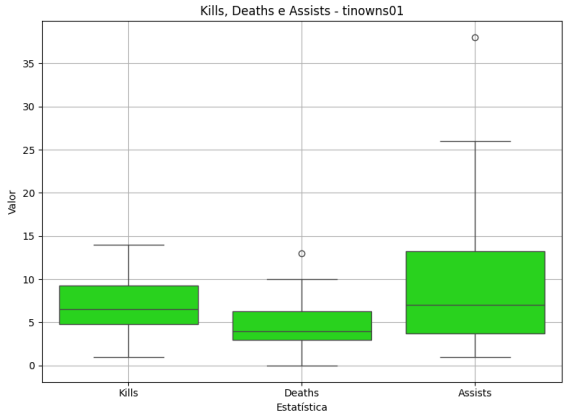


Figura B11 – Boxplot para as variáveis KD, KDA e DR do jogador *tinowns*

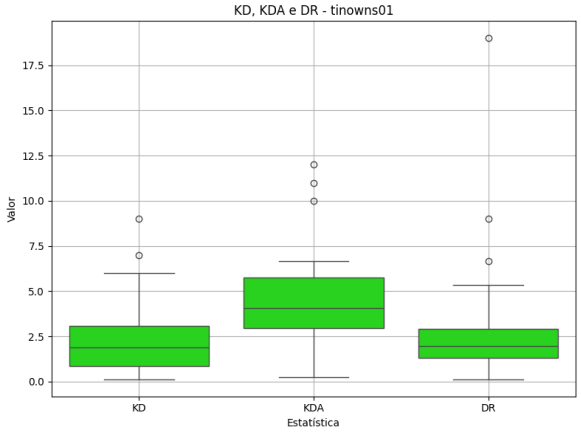


Figura B12 – Boxplot para a variável DF do jogador *tinowns*

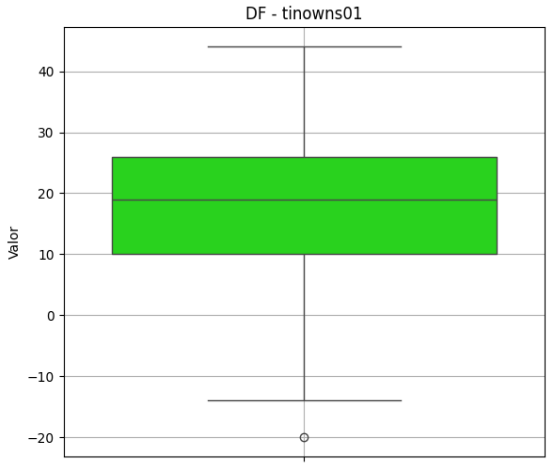


Figura B13 – Histograma para as variáveis Kill, Death e Assists do jogador *TitaN*

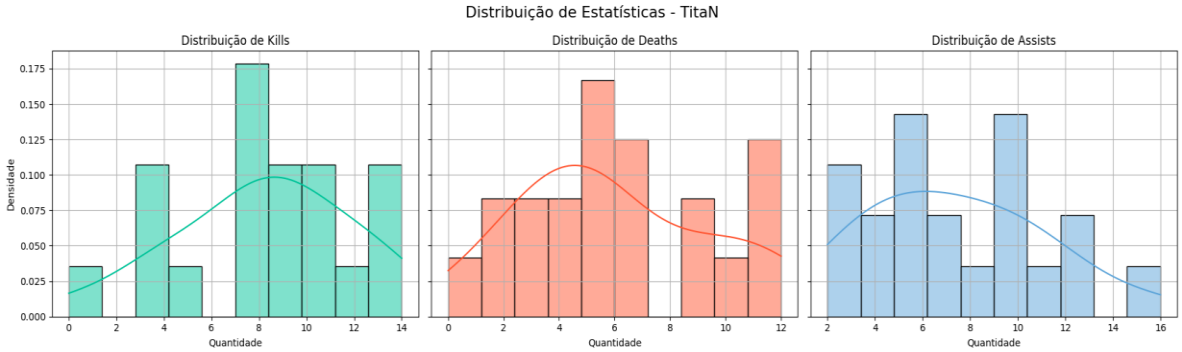


Figura B14 – Histograma para as variáveis KD, KDA e DR do jogador *TitaN*

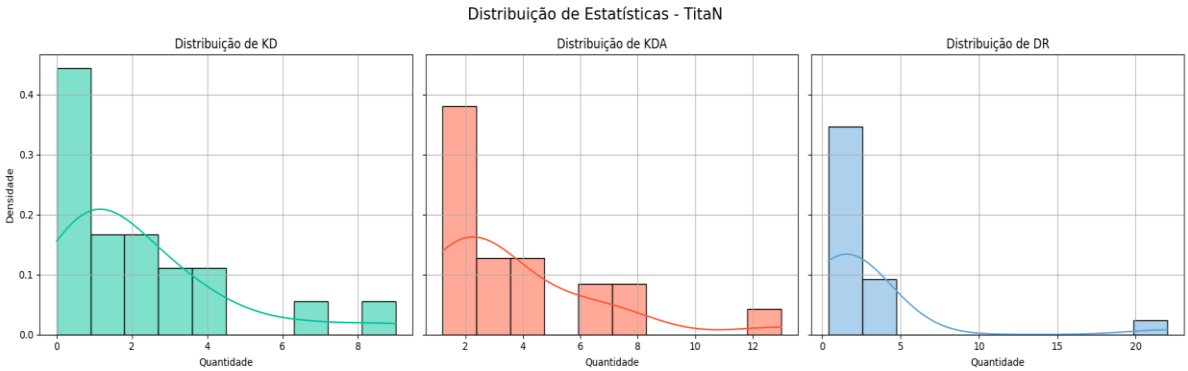


Figura B15 – Histograma para a variável DF do jogador *TitaN*

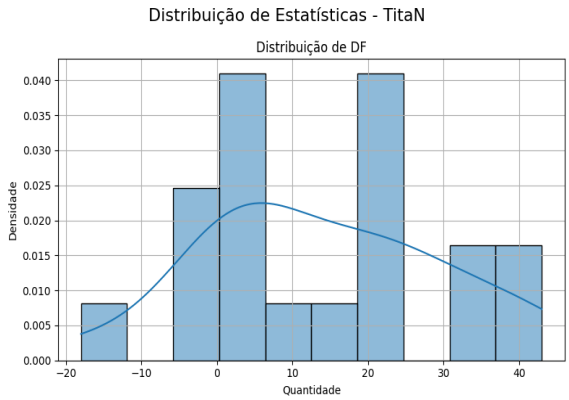


Figura B16 – Boxplot para as variáveis Kill, Death e Assists do jogador *TitaN*

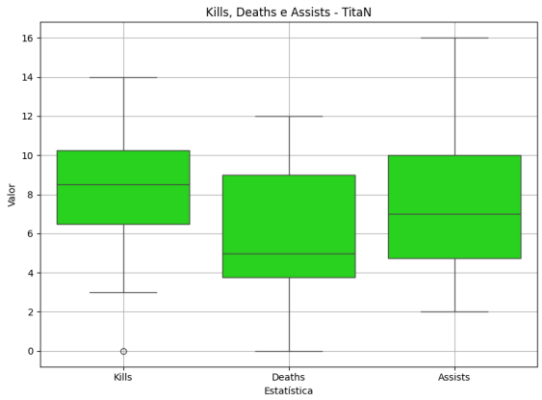


Figura B17 – Boxplot para as variáveis KD, KDA e DR do jogador *TitaN*

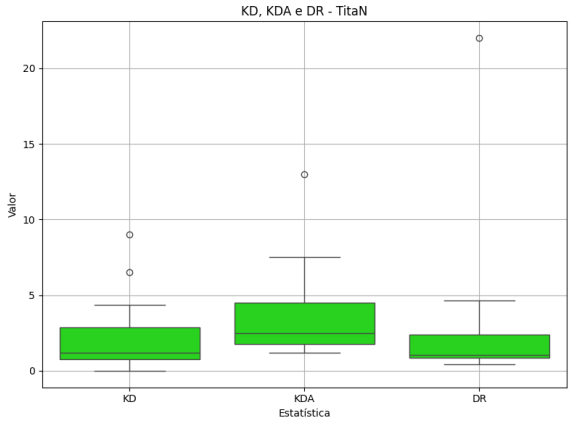


Figura B18 – Boxplot para a variável DF do jogador *TitaN*

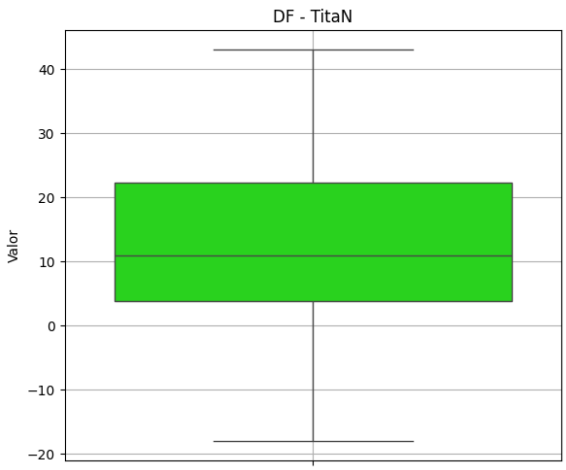


Figura B19 – Histograma para as variáveis Kill, Death e Assists do jogador *dyNquedo*

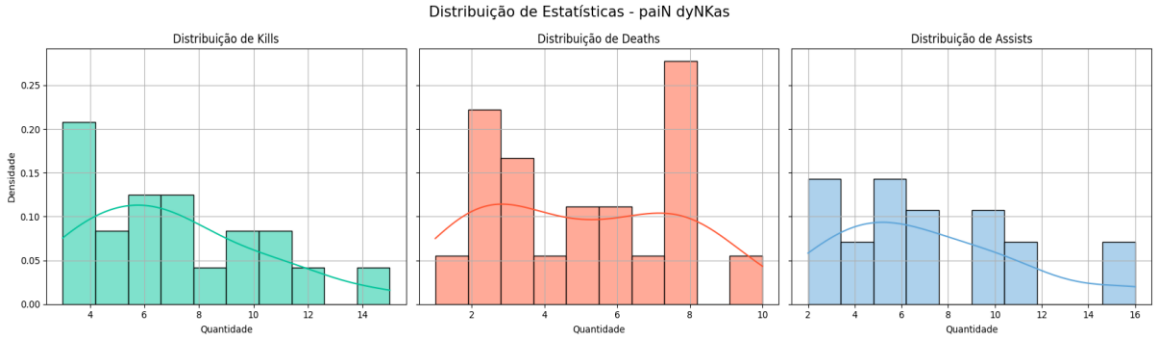


Figura B20 – Histograma para as variáveis KD, KDA e DR do jogador *dyNquedo*

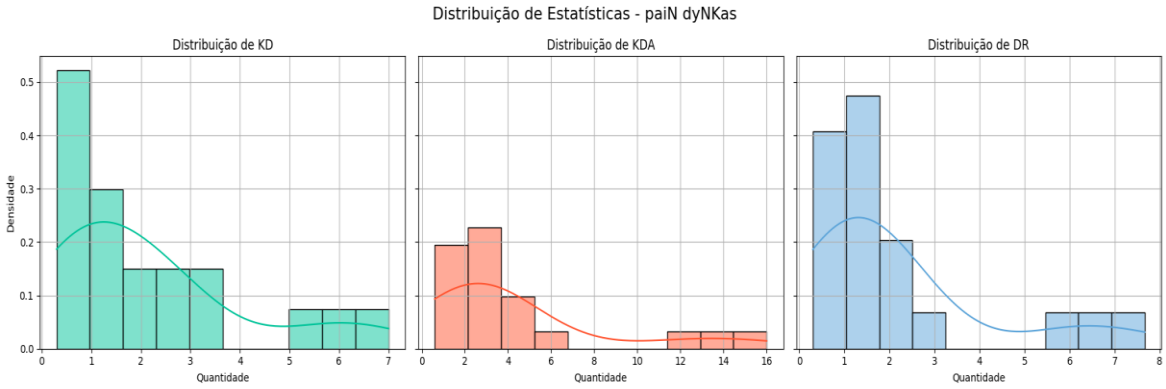


Figura B21 – Histograma para a variável DF do jogador *dyNquedo*

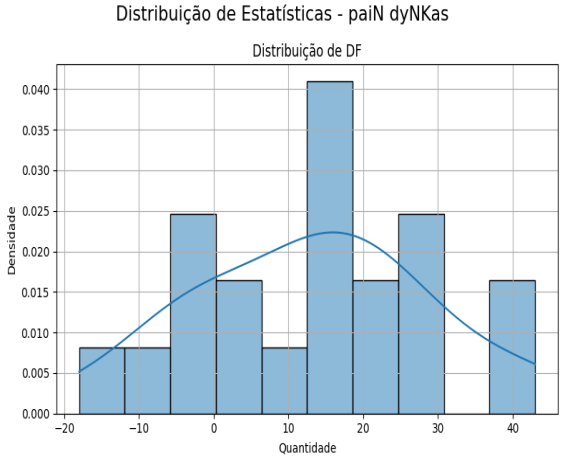


Figura B22 – Boxplot para as variáveis Kill, Death e Assists do jogador *dyNquedo*

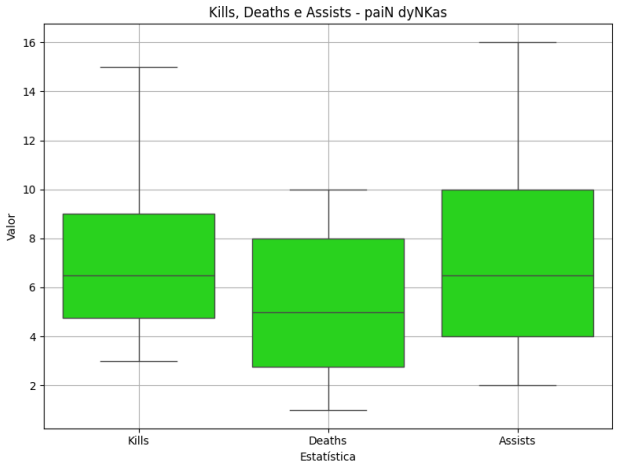


Figura B23 – Boxplot para as variáveis KD, KDA e DR do jogador *dyNquedo*

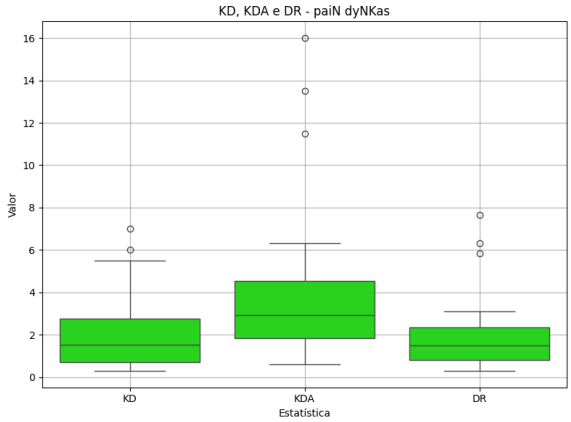


Figura B24 – Boxplot para a variável DF do jogador *dyNquedo*