PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA E DE ARTES
GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

MINERAÇÃO DE DADOS NA BASE DE DADOS DO JOGO *LEAGUE OF LEGENDS*: CONSTRUÇÃO DE MODELOS PREDITIVOS DE VITÓRIAS E DERROTAS DE UMA EQUIPE

AMAURY FLORENTINO FERREIRA

GOIÂNIA
2023

AMAURY FLORENTINO FERREIRA

MINERAÇÃO DE DADOS NA BASE DE DADOS DO JOGO *LEAGUE OF LEGENDS*: CONSTRUÇÃO DE MODELOS PREDITIVOS DE VITÓRIAS E DERROTAS DE UMA EQUIPE

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para obtenção do título de Bacharel em Engenharia de Computação.

Orientadora: Profa. Dra. Solange da Silva.

GOIÂNIA

2023

AMAURY FLORENTINO FERREIRA

MINERAÇÃO DE DADOS NA BASE DE DADOS DO JOGO *LEAGUE OF LEGENDS*: CONSTRUÇÃO DE MODELOS PREDITIVOS DE VITÓRIAS E DERROTAS DE UMA EQUIPE

Este Trabalho de Conclusão de Curso julgado adequado para obtenção de título de Bacharel em Engenharia de Computação, e aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, em ___/___/_____.

Banca examinadora:

_____

Orientadora: Profa. Dra. Solange da Silva

_____

Profa. Dra. Maria Jose Pereira Dantas

_____

Prof. Dr. Jose Elmo de Menezes

GOIÂNIA

2023

AGRADECIMENTOS

Agradeço primeiramente a Deus por me guiar nos momentos difíceis e nos momentos bons. À minha família que nunca deixou de desacreditar em nenhum momento da minha capacidade. Em especial para a minha mãe, Adriana Rodrigues, que me deu suporte em todos os momentos e ao meu pai, Luiz Marcos, que sempre batalhou pelo melhor para a família.

À minha orientadora professora Dra. Solange da Silva que foi fundamental para a elaboração deste trabalho, dando conselhos e orientações e pela grande paciência dedicada. À banca examinadora do TCC2, que avaliou o trabalho e concedeu valiosas dicas de correção. Agradeço também ao Dr. Sibelius Vieira pela participação como avaliador da banca do TCC I, pelas importantes dicas para realização deste trabalho.

Aos meus colegas, que sempre estiveram comigo nos momentos bons e ruins. Finalmente, agradeço a todos que de uma maneira ou de outra contribuíram para que este trabalho tenha acontecido.

RESUMO

O objetivo deste trabalho foi construir dois modelos preditivos para classificar resultados de partida, o primeiro modelo prediz partidas utilizando dados dos 15 primeiros minutos de partida e o segundo modelo analisa os dados ao final do jogo, baseando em dados do desempenho de equipes no contexto do jogo *League of Legends,* usando técnicas de descoberta de conhecimento em base de dados. Uma pesquisa bibliográfica sobre o *League of Legends,* foi realizada com o objetivo de entender e conceituar o contexto do jogo e das técnicas de mineração dos dados. Quantos aos procedimentos técnicos, é uma pesquisa bibliográfica e experimental. O trabalho propôs a utilização dos algoritmos de Regressão Logística, *Random Florest* e Árvore de Decisão. Os resultados obtidos atenderam aos objetivos de pesquisa, desta forma, para o primeiro modelo os algoritmos de Regressão Logística de *Random Florest* tiveram bons resultados de classificação acima do 70% de acurácia, entretanto, a árvore de decisão teve resultado apenas de 64% de classificação. O segundo modelo todos os algoritmos classificaram corretamente os dados acima de 98% de acurácia. Além disso, este estudo proporcionou entender sobre o grau de importância das variáveis para cada modelo, além dos processos de limpeza, transformação, análise de redundância e seleção das variáveis.

Palavras-chave: Modelos Preditivos. *League of Legends.* Mineração de Dados. Algoritmo

ABSTRACT

The objective of this work was to build two predictive models to classify match results, the first model predicts matches using data from the first 15 minutes of the match and the second model analyzes the data at the end of the game, based on team performance data in the context of the match. League of Legends game, using database knowledge discovery techniques. Bibliographic research on League of Legends was carried out in order to understand and conceptualize the context of the game and data mining techniques. As for the technical procedures, it is a bibliographical and experimental research. The work proposed the use of Logistic Regression, Random Forest and Decision Tree algorithms. The results obtained met the research objectives, in this way, for the first model the Logistic Regression algorithms of Random Florest had good classification results above 70% of accuracy, however, the decision tree had a result of only 64% of classification. In the second model, all algorithms correctly classified data above 98% accuracy. In addition, this study provided an understanding of the degree of importance of the variables for each model, in addition to the processes of cleaning, transformation, redundancy analysis and selection of variables.

Keywords: *Predictive Model. League of Legends. Data Mining. Algorithms.*

LISTA DE FIGURAS

TABELAS

QUADROS

LISTA DE ABREVIATURAS

AUC	*Area Under the ROC Curve*
API	*Application Programming Interface*
ARFF	*Attribute-Relation File Format*
CBLOL	Campeonato Brasileiro de *League of Legends*
CSV	*Comma-separated Values*
CRISP-DM	*Cross-Industry Standard Process for Data Mining*
DotA	*Defense of the Ancients*
DCBD	Descoberta de Conhecimento em Base de Dados
FN	*False Negatives*
FP	*False Positives*
FPS	*First Person Shooter*
GEE	*Generalized Estimating Equations*
GHz	GigaHertz
GTX	*Gigar Texel Shader eXtreme*
GUI	*Graphical User Interface*
IA	Inteligência Artificial
JSON	*JavaScript Object Notation*
KDD	*Knowledge Dicovery in Databases*
LoL	*League of Legends*
RAM	Memória de Acesso Randômico
MD	Mineração de dados
MOBA	*Multiplayer Online Battle Arena*
ROC	*Receiver Operating Characteristc*
RTS	*Real-time Strategy*
RTX	*Ray Tracing Extreme*

RP	*Riot Points*
SO	Sistema Operacional
SSD	*Solid State Drive*
TB	*TeraByte*
TN	*True Negatives*
TP	*True Positives*
WEKA	*Waikato Environment for Knowledge Analysis*

SUMÁRIO

1 INTRODUÇÃO

As indústrias dos games é um dos segmentos do entretenimento mais rentáveis do mundo, arrecadando mais que as indústrias dos cinemas e as indústrias musicais somadas. Conforme Pacete (2022), a receita da indústria de games foi de US$ 175,8 bilhões de dólares em 2021. Além disso, estima-se que este mesmo mercado ultrapassará US$ 200 bilhões em 2023, tal sucesso se deve pelo fato de ser um mercado de expansão onde possibilita o crescimento em diversas áreas como dispositivos móveis, consoles e computadores.

Dentro desses segmentos existe o aumento de mercado e público nos esportes eletrônicos conhecido como esportes eletrônicos (eSports). O eSports é o conjunto de mídias sociais, competições organizadas e transmissões. Outro setor que cresce juntamente com o eSports é o de apostas esportivas em jogos eletrônicos, sendo que no Brasil esta é a terceira opção mais famosa entre os brasileiros para fazer suas apostas, mostrando seu grande potencial (BTOBET, 2022).

Um dos eSports populares e mais rentáveis no mundo é o *League of Legends* (LoL), desenvolvido pela *Riot Games*, possuindo uma comunidade com cerca de 124 milhões de contas ativas em 2022 (LUCCA, 2022). Além disso, em 2019 a *Riot Games* informou que o pico diário de jogadores era de 8 milhões simultâneos (MARQUES, 2019).

Em 2021, quatro dos dez maiores torneios de eSports mais assistidos foram relacionados ao jogo *League of Legends*, em especial ao *Worlds* 2021, Campeonato Mundial de LoL que reúne os melhores times de cada região, contabilizando 174,8 milhões de horas assistidas, ficando em primeiro lugar no *rank* (BORISOV, 2021).

League of Legends é um jogo no estilo *Multiplayer Online Battle Arena* ou Arena de Batalha de Multijogadores (MOBA), desenvolvido e publicado pela empresa *Riot Games*, que se caracteriza por um jogo de arena de batalha *online* para vários jogadores (LEAGUE OF LEGEND, 2022).

Nesse contexto, pode-se observar que os MOBAs se tornam jogos competitivos com diversidades nas ações dos campeões durante uma partida (DRACHEN et al., 2014), essa diversidade está entrelaçada ao desempenho individuais dos jogadores e das equipes durante a partida como (ouro ganho, campeões abatidos, neutralizações, dano causado, cura recebida, entre outros).

Pode-se relacionar a competividade dos jogadores com a popularidade dos jogos e essa popularidade está relacionada diretamente com a transmissões de torneios, que fazem jogadores casuais terem comportamento desportistas profissionais (HELLEU et al., 2014). Assim, parte-se de uma visão na qual se tem um jogo com relações complexas, poucos exploradas. Desta forma, um estudo possível é entender relações que impactam na vitória ou derrota de uma equipe (BARCELLOS, 2017).

No contexto de análise de desempenho de uma equipe, esta monografia explora a área de mineração de dados (*data mining*) para tratamento dessas informações coletadas da base de dados da *Application Programming Interface* (API) da *Riot Games*. Desta forma, busca-se entender o comportamento da equipe, de forma a representar, a partir dos dados, quais são as principais características obtidas a partir das técnicas de aprendizado de máquina para tomada de decisão, assim criando um modelo preditivo para vitória de uma equipe em partidas do *League of Legends*.

Justifica estudar este tema porque a mineração de dados é uma área de estudo que está em constante crescimento. O *League of Legends*, assim como os demais jogos, tiveram aumento de jogadores casuais e ganhos de espectadores em transmissões esportivas durante a pandemia, o que possibilitou alta demanda por entretenimento virtual (PACETE, 2022). Assim, é relevante construir um modelo que seja capaz de reconhecer as principais variáveis que fazem a diferença dentro de uma partida. E, que a partir disso, possa prever resultados dos jogos de uma equipe, para fins de descoberta de conhecimento de um jogador casual, para organizações profissionais ou para fins de apostas esportivas.

Diante deste contexto, esse projeto visa responder a seguinte questão de pesquisa: **- Quais fatores influenciam na classificação dos resultados de uma partida baseado no desempenho das equipes no contexto do jogo *League of Legends*, em dois momentos distintos do jogo?**

Este trabalho tem como objetivo geral aplicar dois modelos preditivos para classificar resultados das partidas, o primeiro modelo é baseado no desempenho das equipes aos 15 primeiros minutos e o outro prediz os dados ao final do jogo no contexto do jogo *League of Legends*, usando técnicas de mineração.

Os objetivos específicos são três:

- Investigar critérios de seleção de variáveis no contexto do *League of Legends* em dois momentos da partida: aos 15 minutos e ao final do jogo;

- Aplicar os algoritmos de *machine learning:* regressão logística, *random florest* e árvore decisão;
- Identificar as principais variáveis indicadoras de previsão de resultados de uma equipe para cada algoritmo, visando analisar para cada modelo.

Espera-se que os resultados deste trabalho possam contribuir:

- Com futuras pesquisas de mineração de dados voltadas ao eSports;
- Gerar resultados com uma boa precisão a partir da investigação de critérios de seleção de variáveis aplicado aos modelos;
- Identificando as variáveis mais importantes para uma equipe conseguir uma vitória em dois momentos de jogo: aos 15 minutos e ao final de jogo.

Quantos aos aspectos metodológicos, a natureza desta pesquisa é um resumo de assunto. Quanto aos seus objetivos é uma pesquisa exploratória e descritiva. Em relação aos procedimentos técnicos, é uma pesquisa bibliográfica e experimental.

Esta monografia está estruturada da seguinte maneira: neste capítulo estão apresentados o contexto do trabalho, questão de pesquisa, objetivo geral e específicos, além do resultado esperado. O capítulo 2 traz o referencial teórico, no qual são apresentados conceitos e definições sobre o tema e trabalhos correlatos. O capítulo 3 traz os procedimentos metodológicos para atingir o objetivo geral. O capítulo 4 apresenta o processo de preparação dos dados, dividindo a pesquisas em dois modelos e aplicando técnicas de preparação dos dados. O capítulo 5 descreve os resultados do experimento para cada um dos dois modelos propostos. Finalmente, o capítulo 6 traz as considerações finais sobre o Trabalho de Conclusão de Curso II (TCC II), abordando quais modelos de *machine learning* obteve melhor resultado, além de sugestões de trabalhos futuros para continuidade desta pesquisa.

2 REFERENCIAL TEÓRICO

Este capítulo traz conceitos e definições referentes ao tema, descreve técnicas e ferramentas de mineração de dados no contexto do jogo *League of Legends*, além de descrever o jogo e suas principais características. Além disso, apresenta alguns trabalhos correlatos.

2.1 League of Legends

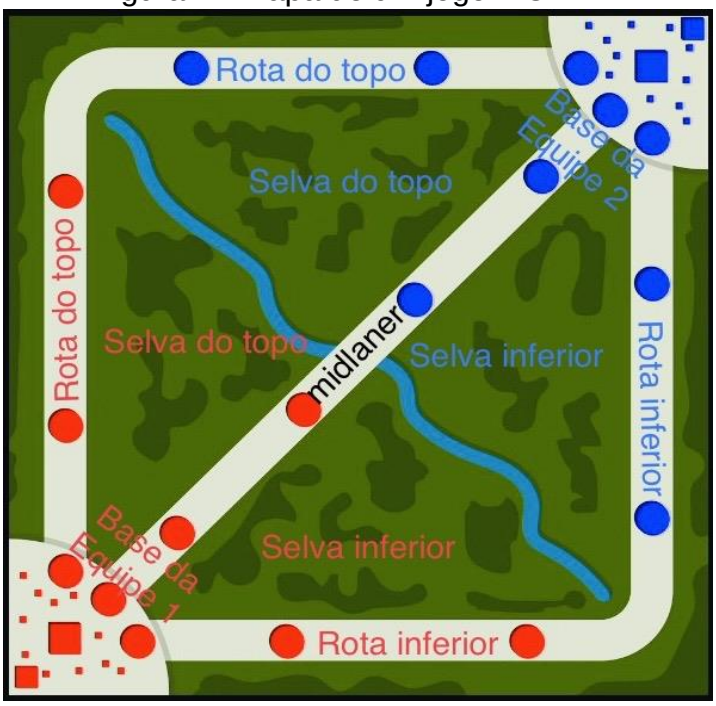
Nessa seção será apresentado inicialmente o contexto histórico dos jogos do gênero MOBA, posteriormente os conceitos do jogo *League of Legends,* exemplificando o funcionamento do jogo. Finalmente, será retratada a importância do jogo no mundo dos eSports e das apostas esportivas.

2.1.1 MOBAs

Em geral, jogos no estilo MOBA têm suas origens nos jogos no gênero *Real-time Strategy* (RTS). Apesar de serem jogos de características diferentes de outros gêneros, os jogos MOBA se mostram bastante similares em jogabilidade e similaridades entre si. O sucesso dos MOBAs se deve por conta da popularidade dos jogos de estratégias. Houve grande interesse por parte dos jogadores de criarem modos de jogos personalizados denominados de "*mods".* Desde o início do desenvolvimento desse gênero tem-se a ideia de um jogo em tempo real de ambiente praticamente observável no qual é possível controlar um personagem denominado "herói" ou "campeão", escolher uma rota e avançar as tropas e conquistar a base inimiga (SILVA, 2016; NASCIMENTO Jr., 2017).

Para Silva (2016), o MOBA de modo geral é um jogo que exige um planejamento e rápida tomada de decisão definidos a curto e a longo prazo, os jogos do gênero MOBA são classificados como não determinísticos devido às probabilidades presentes dentro de uma partida do jogo e não persistentes, pois a cada partida o jogo volta em seu estado inicial. Na Figura 1 é apresentada a generalização de um mapa no MOBA, normalmente é composto pela selva, onde é possível retirar recursos adversários e pelas três rotas: rota do topo, rota do meio e rota inferior. O mapa é dividido entre dois times e uma área comumente chamada de rio, sendo possível avançar através da destruição de torres.

Figura 1 - Mapa de um jogo MOBA



Fonte: adaptado de Gomes, 2022.

O primeiro MOBA de sucesso foi desenvolvido do jogo *Starcraft* da desenvolvedora *Blizzard*, denominado de *AeoN of Strife e* foi um dos pioneiros do estilo de jogabilidade e muito bem aceito pela comunidade (SILVA, 2016). Segundo Minotti (2014), com o sucesso de *AeoN of Strife* a própria *Blizzard* disponibilizou em seu jogo *WarCraft III* lançado em 2002, algumas melhorias relacionadas ao antigo jogo *Starcraft,* como cenários diferentes, melhorias tridimensionais, raças diferentes e melhorias no ambiente de personalização. Nesse momento, com a aceitação da comunidade de *modder,* voltou-se a atenção e a dedicação para a criação de um dos mapas mais populares de todos os tempos, o *Defense of the Ancients* popularmente conhecido como DotA (MINOTTI, 2014).

O sucesso do DotA proporcionou às empresas apostar em jogos desse gênero. Em 2009 a *Riot Games* lançou o seu jogo *League of Legends,* posteriormente em 2013 a *Valve lançou* seu jogo baseado no DotA, uma nova versão intitulada de *Dota2 (SILVA, 2016).* Outros games do gênero de destaques na atualidade são: *Heroes of the Storm*, da *Blizzard; Smite,* da *LevelUP!; Mobile Legends: Bang Bang,* da *Moonton* e *League of Legends: Wild Rift*, da *Riot Games.*

Atualmente, vale ressaltar o aumento por procura de jogos do gênero MOBA para dispositivos móveis, mostrando que também é possível explorar esse nicho (FLOTER,

2021). Destaque para os jogos *Mobile Legends: Bang Bang* e o *League of Legends Wild Rift*, versão *mobile* do jogo *League of Legends*.

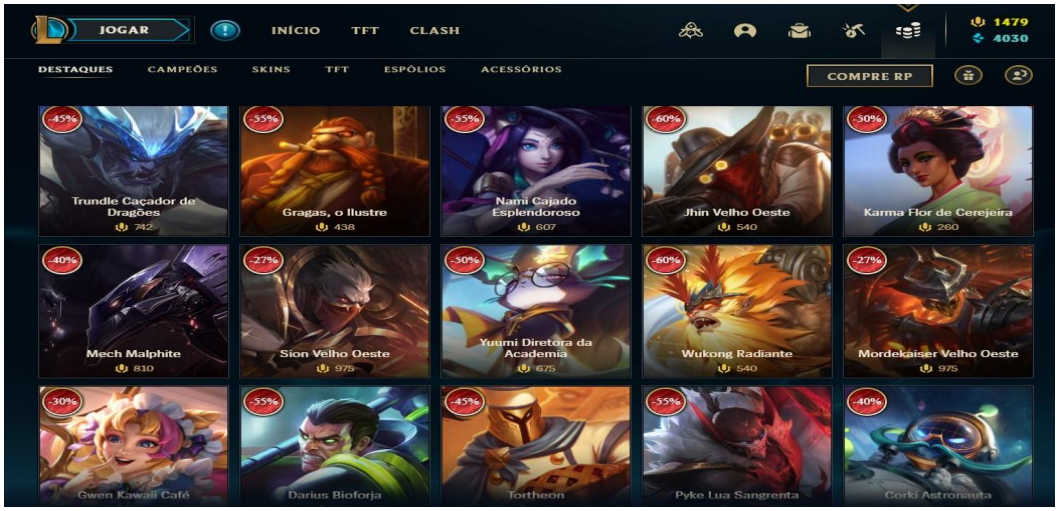
2.1.2 *League Of Legends*

Esta seção será descrita conforme o material disponível em *League of Legends* (2022). *League of Legends* conhecido também como LoL é um jogo eletrônico do gênero MOBA (*Multiplayer Online Battle Arena) f*oi desenvolvida pela *Riot Games* para Windows e Mac Os X. Este jogo está no formato *free-to-play*, ou seja, gratuito. No entanto, se algum jogador optar por um personagem específico de imediato sem conseguir as essências azuis: moedas que são adquiridas através de missões e partidas no jogo para compra de campeões; ele poderá utilizar os *Riot Points (*RP) que é comprado através de compra monetária com dinheiro real.

Outro ponto importante que faz o *League of Legends* um grande sucesso de mercado é a utilização de RP para compra de passe de batalhas e *skin*, que nada mais é que um visual alternativo de um campeão. É importante ressaltar que a utilização de investimento monetário não proporciona vantagens imensas dentro do campo de batalha, apenas acelera o processo de desenvolvimento do jogador no jogo (MEZZOMO, 2014).

A Figura 2 mostra como é o formato da loja do *League of Legends*, mostrando algumas *skin* de promoção. Além disso, no canto superior direito é possível observar as duas moedas, primeiro a *Riot Points* e segundo as moedas de essências azuis:

Figura 2 – Interface da loja do cliente do *League of Legends*

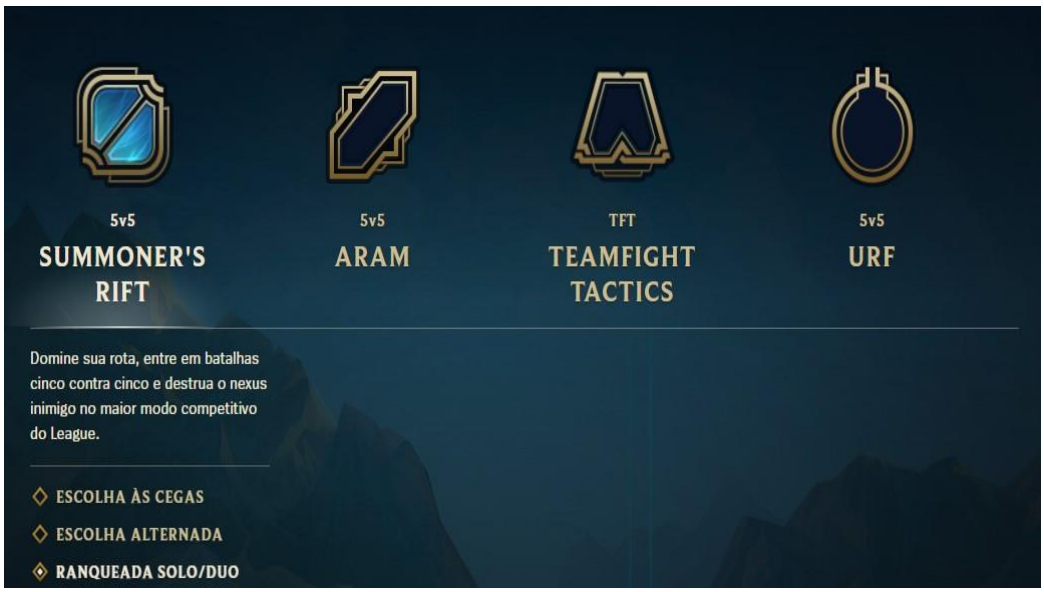


Fonte: adaptado de *League of Legends*, 2022.

O LoL atualmente, considerando o ano de 2022, conta com 4 modos de jogos: *Summoner's Rift, Howling Abys* (ARAM), *TeamFight Tactics* (TFT) e Ultra rápido e Furioso (URF). Levando em consideração que cada mapa é diferente totalmente um do outro, nesta pesquisa será utilizado apenas o modo de jogo *Summoner's Rift* em partidas ranqueadas.

A Figura 3 mostra os modos de jogos que é possível escolher dentro do cliente do LoL:

Figura 3 - Modos de jogo do LoL



Fonte: adaptado do League of Legends, 2022.

Summoner's Rift é o mapa mais jogado do jogo, entre seus objetivos principais cada equipe começa na sua base. O princípio é proteger o seu *Nexus* e atacar o *Nexus* inimigo. Quem conseguir o feito automaticamente conseguirá a vitória (SCOLA, 2022).

Uma partida se caracteriza por um cenário (*Summoner's Rift*) em que duas equipes, geralmente cada qual com cinco jogadores, lutam entre si, sem limite de tempo, visando destruir a base inimiga. Os jogadores são conhecidos como invocadores e cada um tem o direito de escolher um entre os mais de 160 personagens disponíveis no jogo, conhecido como "herói" ou "campeão" com habilidades e um kit único para derrotar invocadores do time inimigo.

Durante a partida, os personagens vão coletando ouro de diversas formas, ou seja, com abates de personagens, *minions*, destruição de torres ou por tempo de jogo que são usados para comprar itens e melhorar seus atributos e habilidades; outro item a avaliar são os pontos de experiência conseguidos através de abates de tropas ou personagens e destruir estruturas de torres inimigas.

Todos os campeões desse modo começam no nível um e à medida que o jogo se desenvolve, os seus campeões aumentam seus níveis até o máximo de 18. As partidas desse modo duram em média, entre 20 e 45 minutos. Além disso, o jogo também possui assistência de estruturas de defesas, unidades controladas por Inteligência Artificial (IA), que se caracterizam pelos *minions*, além de objetivos neutros: Arauto, Baron e Dragão.

Objetivos neutros são caracterizados por locais onde se encontram grande monstros que possuem grande recompensa. Também são locais onde ocorre com certa facilidade batalhas de equipes por disputa de objetivo. A Figura 4 exemplifica o mapa de *Summoner's Rift,* mostrando as rotas (topo, selva, meio, rota inferior), nexus, base e, por fim, os objetivos neutros:

Figura 4 - Mapa de Summoner's Rift

Fonte: Novais & Tartaglia, 2021.

Segundo Scola (2022), no League of Legends é possível dividir os 5 jogadores em 4 rotas conforma a Figura 4, onde cada um deles exerce uma função. Essas funções são descritas seguinte forma:

- **Topo**: jogador que fica posicionado na parte superior do mapa (rota Topo), geralmente optam por bonecos de bastante vida que são capazes de resistir por muito tempo nas batalhas;
- **Caçador**: jogador que cuida da selva e dos objetivos, além de ser o elemento surpresa para que sua equipe consiga vencer a fase de rotas;
- **Meio**: o jogador da rota do meio deve ser capaz de carregar a sua equipe, ajudando no dano constante;
- **Atirador**: jogador da rota inferior que tem como objetivo aplicar a maior quantidade de dano nas lutas;
- **Suporte**: o suporte deverá proteger seus aliados e ajudar na criação de jogadas. Durante a fase de rotas o jogador começa pela rota inferior, ajudando o atirador e criando vantagem pelo mapa.

Alguns pré-requisitos dispostos para a vitória são: é necessário que ao menos cinco das onze torres inimigas sejam destruídas e um dos três inibidores. Vale ressaltar que os inibidores, após 5 minutos são restaurados, portanto é necessário que ele seja destruído novamente para que consiga derrotar o Nexus.

Geralmente as partidas são divididas em três fases distintas, de acordo com o desempenho dos jogadores e tempo da partida:

- ***Early Game*:** também conhecida como fase de rotas, acontece entre os 0 a 15 minutos de partida. O objetivo principal dessa fase é conseguir o máximo de recursos possíveis;
- ***Mid Game**:* acontece entre os 16 a 30 minutos. Nessa fase acontece diversas lutas por objetivos e até mesmo o fim do jogo, caso a equipe consiga bastante vantagem no *Early Game.*
- ***Late Game**:* após os 30 minutos entramos na fase final do jogo *Late Game.* Nessa fase, qualquer batalha errada por objetivo pode transformar na vitória ou derrota de uma equipe.

No *League of Legends*, além dos desempenhos dos jogadores e tempo de partida influenciarem nas fases citadas acima, os campeões também têm grande influência.

Uma vez escolhido o campeão, alguns campeões possuem pico de poder no *Late Game*, isto é, esse campeão necessita que o *Early Game e o Mid Game* sejam tranquilos para que consiga chegar no pico de poder máximo do campeão. Da mesma forma existem campeões que no começo de partida são extremamente fortes, uma vez que, o campeão não consiga muita vantagem no começo de partida ele decairá ao decorrer do tempo (LEAGUE OF LEGENDS, 2022).

2.1.3 eSports

Os esportes eletrônicos ou popularmente conhecido como eSports, são competições de jogos eletrônicos em que atletas profissionais disputam partidas de forma presencial ou on-line (GE, 2021). Segundo o site Gazeta Esportiva (2021), existem diversos gêneros de jogos que aderem ao eSports, os mais populares são: MOBA *(Multiplayer Online Battle)*, FPS *(First Person Shooter), Card Games, Battle Royale, Simuladores, Evo Fighting games*.

De modo geral os eSports vem em constante crescente. Segundo estimativas o mercado de eSports poderá atingir em 2022 um público de 532 milhões de pessoas, tendo um crescimento de +8,7% ao ano e gerando uma receita mundialmente de US$ 1,38 bilhões até o final de 2022 (TRISTÃO, 2022). Esses games são de fáceis transmissões, seja nos canais de TV Fechada no Brasil (SportTV, TNT Sports) ou seja por canais de *Streaming* (*TwitchTV, YouTube, Facebook Gaming*).

No Brasil, com o notório sucesso e visibilidade dos eSports, um dos principais campeonatos de eSports assistido aqui no brasil é o campeonato brasileiro de *League of Legends (CBLOL).* O CBLOL é composto por 8 times, no qual, o vencedor do campeonato no 2ª split ganha uma vaga para o mundial do LoL que ocorre uma vez por ano*.* Segundo Vasquez (2022), a final do CBLOL 2022 teve um pico de 331 mil espectadores. Na final do no mundial de LoL de 2022, ocorrida no início de novembro, teve um pico por volta de 1 milhão e 400 mil espectadores (PERREIRA, 2022).

Para que a relação entre o eSports e o *League of Legends* seja melhor compreendida primeiro deve-se entender as partidas ranqueadas. No LoL é possível jogar partidas ranqueadas quando sua conta atinge o level trinta e tenha no mínimo 20 campeões liberados (CARBONE, 2021). Após esses requisitos é possível participar do sistema de classificação, no qual, os jogadores são divididos em *Tiers*: Ferro, Bronze, Prata, Ouro, Platina, Diamante, Mestre, Grão-Mestre e Desafiante.

Os jogadores que se destacam, isto é, conseguem atingir o *Tier* Desafiante - passam a concorrer em campeonatos oficiais do jogo. Por conta disso, é possível disputar premiações cada vez maiores, muito se deve pela popularização do jogo e do crescente números de eventos (MEZZOMO, 2014). De acordo com o site eSports *charts* (2021), o *League of Legends* conseguiu o sexto lugar no rank de jogos com maiores premiações, com pouco mais de 8 milhões de dólares em premiação em 2021.

Entretanto de todos os jogadores que participam de ranqueadas competitivas, apenas 0,18% são do *tier M*estre, 0,026% do *tier* Grão-Mestre e 0,011% representam o tier Desafiante (MILELLA, 2022). A Tabela 1 mostra a relação entre a classificação e a porcentagem de jogadores nas filas *solo-queue* de novembro de 2022:

Tabela 1 - Relação entre o *Tier* e a porcentagem

Classificação	**Percentagem**	**Classificação**	**Percentagem**	**Classificação**	**Percentagem**
Ferro IV	0,37%	Prata II	8,1%	Diamante IV	0,70%
Ferro III	0,58%	Prata I	6,5%	Diamante III	0,33%
Ferro II	1,1%	Ouro IV	12%	Diamante II	0,44%
Ferro I	2,0%	Ouro III	5,1%	Diamante I	0,26%
Bronze IV	5,5%	Ouro II	4,5%	Mestre	0,18%
Bronze III	5,2%	Ouro I	3,1%	Grão-Mestre	0,026%
Bronze II	6,7%	Platina IV	5,6%	Desafiador	0,011%
Bronze I	7,0%	Platina III	1,9%		
Prata IV	11%	Platina II	1,4%		
Prata III	7,2%	Platina I	1,8%		

Fonte: adaptada de Milella, 2022.

Como pode ser observado na Tabela 1, há uma grande discrepância na qual existem equipes que buscam grandes jogadores para compor seu time. Entretanto, é notório que existe poucas pessoas se destacando no jogo. Segundo o site Ge (2021), este é um mercado que está em constante crescimento e a tendência é evoluir para os próximos anos.

2.1.5 Apostas Esportivas

Com a popularidade do eSports, também houve aumento relativo ao número de apostas esportivas no meio eletrônico. Atualmente no ano de 2022, esta modalidade se posiciona-se como terceira colocada como mais apostas, perdendo apenas do futebol e do basquete (VICTOR, 2022).

De acordo com o site Grande Prêmio (2022), alguns fatores que impulsionam o crescimento das apostas esportivas são:

- mesma experiência que os esportes tradicionais;
- assistir aos eventos ao vivo e online;
- diversidade de oferta de gênero como MOBA, FPS, Battle Royale etc.

O *League of Legends,* por exemplo, possui diversões campeonatos com transmissões gratuitas nas quais se pode apostar e assistir ao jogo através do *streaming*. Além disso, o LoL possui uma variedade interessante de apostas, são elas: vitória de uma equipe, quantidade de abates, destruição de torres, destruição de objetivos neutros (barões, dragões, arauto), primeira *kill*, ouro ganho, *handicap* dentre outros.

Vale ressaltar que essas apostas podem variar dependendo da casa de aposta. Por fim, parte-se da ideia que seria interessante um modelo classificador, capaz de predizer vitória e derrota para uma equipe para fins de apostas esportivas, tomada de decisão de organizações profissionais e para melhorias de jogadores casuais no âmbito do *League of Legends.*

2.2 Descoberta de conhecimento em base de dados

Nesta seção é apresentada os conceitos de Descoberta de Conhecimento em Base de Dados (DCBD), exemplificando as fases que compõem o DCBD, além de conceitos de mineração de dados, técnicas, métricas de avaliação e ferramentas que vão ser utilizadas no trabalho.

Em meio ao avanço da tecnologia, dados são gerados constantemente, em que a capacidade de armazenamento de dados não é equivalente à capacidade de analisar e extrair informações úteis desses dados. Dentro desse cenário surge a Mineração de Dados com a finalidade de encontrar padrões, anomalias, correlações em grandes bases de dados e auxiliar na descoberta de conhecimento. Vale ressaltar

que a mineração de dados é um dos processos que compõem o processo de DCBD (SILVA; PERES & BOSCARIOLI, 2016).

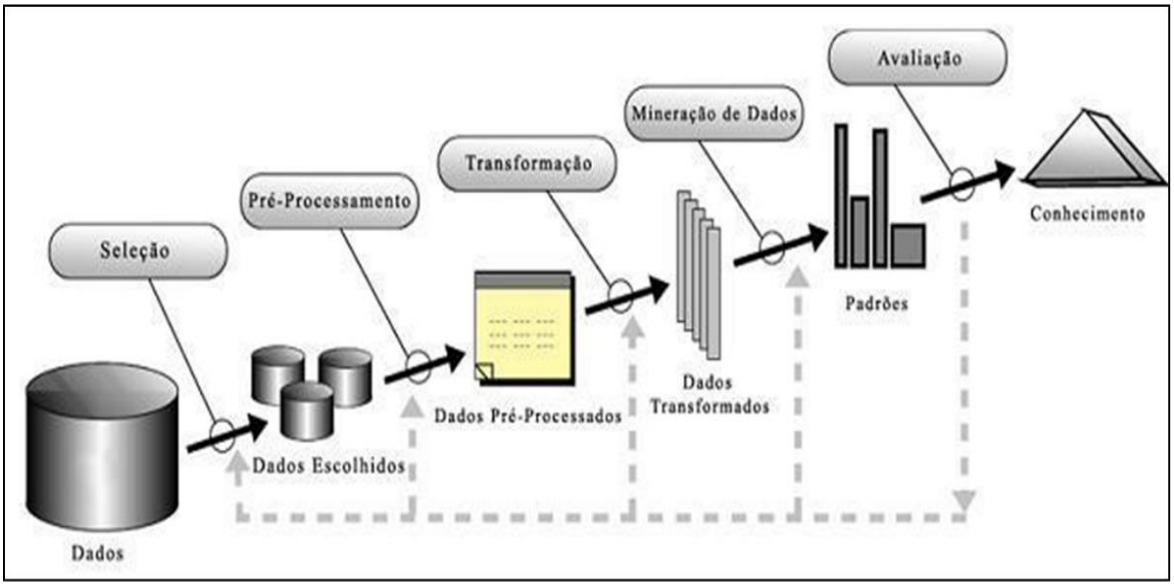
A mineração de dados é uma parte de um processo extenso conhecido como Descoberta de Conhecimento em Bases de dados, conhecido também como *Knowledge Discovery in Databases* (KDD), popularmente conhecido como o processo de KDD (CASTRO & FERRARI, 2016). Este processo pode ser entendido como sendo a conversão de dados brutos em informações úteis, através de uma série de transformações até que seja possível analisar os resultados da mineração de dados (PINHEIRO, 2008).

2.2.1 Etapas do Processo de DCBD

O KDD é composto por etapas que contêm sequência de passos que auxiliam na busca por informações e conhecimentos de forma não-trivial de uma base de dados. Cada uma dessas etapas possui uma interseção com as demais, desenvolvendo atividades específicas e melhorando os resultados (RELICH & MUSYNSKI, 2014).

Esse processo recebe uma entrada de dados brutos e retorna na saída um conhecimento útil sendo dividido em 5 etapas ao longo do processo de KDD: seleção de dados; pré-processamento dos dados; transformação dos dados; mineração de dados e interpretação e avaliação dos resultados. A Figura 5 ilustra esse processo, representando suas várias fases:

Figura 5 - Processo de Descoberta de Conhecimento em Base de Dados (KDD)



Fonte: Fayyad et al., 1996.

Cada etapa mostrada na Figura 5 será explicada a seguir:

- **Seleção dos Dados:** conhecido como etapa de “Redução de Dados” segundo Rosa (2017), é a primeira etapa de aprendizado do domínio da aplicação contendo todas as possíveis variáveis e registros que se pretende analisar. Vale ressaltar que esse conjunto de dados pode ou não estar estruturado, além disso os dados coletados podem vir de banco de dados, planilhas ou até mesmo dados escritos ou digitados (MOURA, 2019).

- **Pré-processamento e Limpeza dos Dados:** consiste na escolha de técnicas que sejam capazes de excluir dados redundantes e inconsistentes, dados incompletos e, por fim,ok avaliar dados discrepantes através do uso de agrupamentos, regressão, interpolação e inferência. De acordo com Silva; Peres & Boscarioli (2016) a limpeza de dados tem a finalidade de resolver a presença de dados ausentes e a presença de valores ruidosos ou inconsistentes. Além disso, é fundamental que neste processo seja verificada a possibilidade de diminuir o número de variáveis para aquele determinado domínio, analisando durante o processo de KDD para um melhor desempenho dos algoritmos analisados (PADHY; MISHRA & PANIGRAHI, 2012).

- **Transformação dos Dados:** na etapa de transformação dos dados é realizada a integração dos dados. Segundo Silva; Peres & Boscarioli (2016) o objetivo da integração é unificar diferentes fontes de dados em apenas uma base de dados. Além disso, deve-se formatar os dados para que fique adequados para o processo de mineração de dados. Essa integração visa minimizar a presença de valores inconsistentes e redundante de atributos para a transformação dos dados, no qual seja possível, no final deste processo, converter um conjunto de dados bruto em uma forma padronizada, de uso nas fases posteriores do processo de KDD (GOLDSCHMIDT & PASSOS, 2005).

- **Mineração dos Dados:** A etapa de mineração é responsável pela aplicação de técnicas e algoritmos de mineração de dados que possam verificar hipótese e consiga extrair padrões de forma automatizada através dos dados definidos na

etapa anterior. A etapa de mineração de dados é a mais importante do processo de KDD (GOLDSCHMIDT& PASSOS, 2005). Além disso, nessa etapa é possível aplicar o processo de *data mining* diversas vezes. Por fim, se o objetivo não for alcançado, essa etapa pode ser refeita utilizando outras técnicas (CASTRO & FERRARI, 2016).

- **Interpretação e Avaliação dos Resultados:** a etapa de interpretação e avaliação dos resultados também é conhecida como pós-processamento (ROSA, 2017). Nesta etapa os resultados da fase de mineração de dados são interpretados e o desempenho das técnicas utilizadas avaliadas (FAYYAD et al., 1996). A avaliação dos resultados pode ser feita utilizando medidas estatísticas geradas, além disso, caso os resultados não seja o desejado é possível retornar a qualquer uma das etapas citadas anteriormente.

2.2.2 Mineração de dados

Goldshmidt & Passos (2005) definem a Mineração de Dados, ou no inglês, *Data Mining,* como uma das etapas da Descoberta de Conhecimento em Base de Dados. Esse processo pode ser entendido como a parte que desfruta do auxílio dos algoritmos computacionais, utilizado após a extração e limpeza dos dados, com o intuito de descoberta de padrões e análise de tomada de decisão (CABENA et al., 1998).

O crescimento de grandes volumes de dados vem emergindo em diversas áreas das indústrias e pesquisas, são exemplos: bioinformática, análise de redes sociais, visão computacional e jogos digitais (CANOSSA, 2016).

A mineração de dados pode ser aplicada em diversas fontes de dados distintas para previsão ou descrição de informações uteis, como por exemplo, a análise de crédito para clientes. Os bancos podem vir a sofrer com golpes de crédito e o não pagamento de dívidas, uma forma de minimizar esses prejuízos é aplicando técnicas de mineração e dados para classificar os clientes. Dantas et al. (2008), afirmam que os dados passam por uma avaliação e interpretação para chegada de uma conclusão válida e decisiva para o problema.

Segundo Silva (2008), a mineração de dados pode se relacionar com diversas áreas de conhecimento. As principais áreas que envolvem diretamente com a mineração de dados são:

- Banco de Dados: usado para manipular os dados dentro da base de dados, podem vir de diversas formas estruturados ou não;
- Estatísticas: são utilizados métodos estatísticos para avaliar e validar os resultados;
- Inteligência artificial: Utilização de técnicas de aprendizado de máquina para descoberta de padrões através de algoritmos. Os aprendizados de máquinas podem ser classificados em dois grupos: aprendizado supervisionado e não supervisionado.

2.2.3 Tarefas e técnicas de mineração de dados

A utilização de tarefas e técnicas de mineração de dados depende das etapas de pré-processamento e pós-processamento. O pré-processamento transforma os dados brutos em formato apropriado para a realização da mineração através dos algoritmos. E o pós-processamento avaliam o modelo criado, validando os resultados úteis e incorporando ao sistema (MOURÃO, 2018).

O processo de mineração de dados pode ser classificado em quadro etapas, são elas: escolha da tarefa de mineração de dados; escolha da técnica de mineração de dados; escolha do algoritmo e aplicação do processo de mineração de dados (BATISTA, 2003).

As tarefas de mineração de dados são divididas em duas categorias, as tarefas preditivas e as tarefas descritivas. As tarefas preditivas visam predizer valores futuros através de dados já conhecidos. As tarefas de classificação e regressão fazem parte da análise preditiva. As tarefas descritivas encontram padrões para descrever os dados. As tarefas de agrupamento, modelagem de dependências, sumarização e detecção de desvios são exemplos de tarefas descritivas (CASTRO & FERRARI, 2016).

A classificação é considerada a tarefa mais comum entre as outras técnicas de mineração de dados. Ela consiste em dividir os itens em categorias ou classes de destino para prever o que pode ocorrer dentro de uma classe. Por exemplo, é possível utilizar essa tarefa para compra e vendas de ações, conseguindo antecipar ao mercado e minimizando as perdas. Essa tarefa utiliza duas etapas, a primeira denomina-se fase de treinamento do modelo, no qual, o modelo é gerado através dos dados já classificados. Em seguida, ocorre a fase de teste com um conjunto de dados

que não foram usados na etapa anterior, assim é possível verificar a capacidade do modelo fazer previsões corretas de dados (HAN & KAMBER, 2001).

Segundo Han & Kamber (2001), a regressão tem como saída valores numéricos, no qual, é possível determinar a função que representa por meio de equações estatísticas. Goldshmith & Passos (2005) definem a regressão por uma busca por funções lineares ou não, que consigam mapear os registros com valores reais. De maneira semelhante à tarefa de classificação, a regressão também é uma tarefa que segue o modelo de aprendizagem supervisionada, separando dados para o treinamento e testes. Na regressão ocorre o cálculo da distância entre a saída esperada e a saída estimada, tendo como resultado a precisão da predição (CASTRO E FERRARI, 2016).

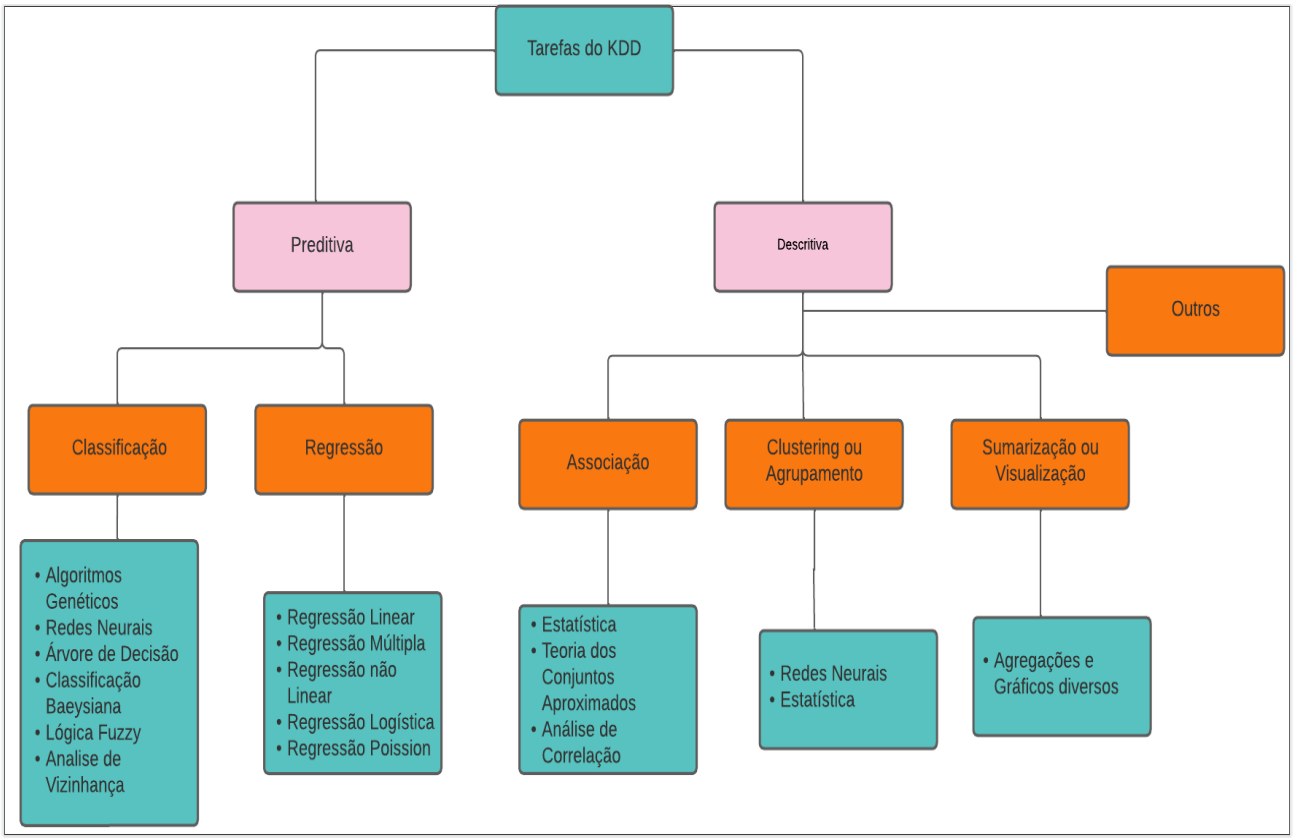
Tarefas de associações tem por objetivo identificar as relações que os atributos têm em dados transacionais, com o intuito de reconhecer padrões (SILVA; PERES & BOSCARIOLI, 2016). Exemplificando, pode-se evidenciar a análise do carrinho de supermercado, em que, pessoas que compram cerveja, também compram fraldas, sendo assim, indicando que apesar da forte relação entre os atributos, não existe qualquer relação entre os produtos.

Clusterização ou agrupamento são tarefas que dividem a base de dados em agrupamento (*clusters)*, de modo que os elementos similares fiquem no mesmo grupo. Segundo Silva, Peres & Boscarioli (2016), são realizadas buscas por similaridades, nas quais é analisada a distância de similaridade e a diferença entre o objeto. Sendo assim, objetivos que possuírem distância de similaridade menores, são agrupados em um mesmo *cluster.* A tarefa de agrupamento é semelhante à tarefa de classificação, no entanto, esta tarefa é classificada como não supervisionada, pois os rótulos de classes são desconhecidos, fazendo com que dependa do algoritmo de agrupamento para descobrir classes aceitáveis (NASCIMENTO Jr., 2017).

A tarefa de sumarização consiste em encontrar uma descrição simples para um conjunto de dados, transformando o conjunto de dados original em uma versão menor (CARDOSO, 2017). Essa tarefa deve ser de forma clara, compreensível e compactada, sendo possível, a implementação através de algoritmos genéticos.

Como visto acima, existem uma vasta gama de técnicas de mineração de dados, sendo possível usar de acordo com o problema que se deseja resolver. Na Figura 6 é apresentado um fluxograma mostrando as principais técnicas e tarefas de mineração de dados.

Figura 6 - Relação entre técnicas e tarefas de mineração de dados



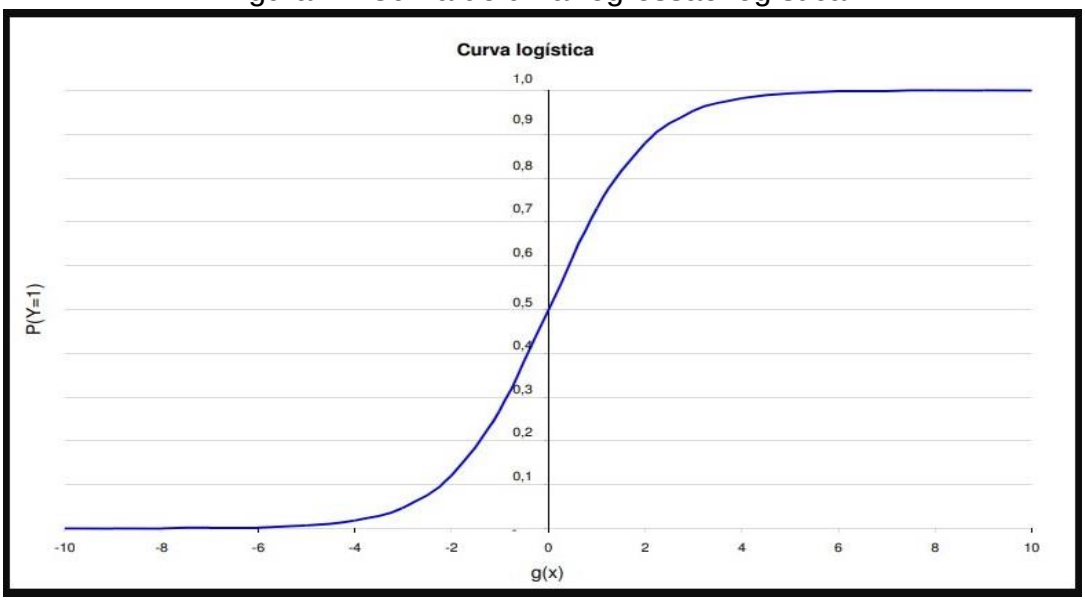
Fonte: autoria própria.

2.2.4 Regressão Logística

A regressão logística é uma das técnicas de modelos preditivos que utiliza matemática para encontrar relações entre dois fatores de dados. Uma das características este algoritmo é o fato da sua variável dependente categórica, ser binária ou dicotômico.

Desta forma, é possível ter representações 0 ou 1, falso ou verdadeiro, sim ou não, dentre outros. A Figura 7 apresenta a visualização de uma curva logística gerada.

Figura 7 - Curva de uma regressão logística.



Fonte: Miguel, 2020.

Miguel (2020), define os principais ingredientes para a regressão logística como: a função de resposta logística e a função de *Logit*. O modelo de regressão logísticas pode ser descrito da seguinte forma:

$$P(Y = 1) = \frac{1}{1 + e^{-g(x)}} \quad (1)$$

onde, *g(x)* é igual:

$$g(x) = B_0 + B_1X_1 \cdots + BpXp \quad (2)$$

A Equação (1) garante que *P* permaneça entre 0 e 1. Segundo Hosmer & Lemeshow (1989), os coeficientes B_0, B_1..., *Bp* são estimados no conjunto de dados a partir do método de máxima verossimilhança, em que, são encontradas combinações de coeficiente que maximiza a probabilidade. Nas Equações 3 e 4 são representadas a variação dos valores de *X* para combinações dos coeficientes B_0, B_1..., *Bp* para o exemplo de curva logística da Figura 7.

Quando

$$g(x) \rightarrow +\infty, então\ P(Y = 1) \rightarrow 1 \quad (3)$$

Quando

$$g\,(\,x\,) \rightarrow\, -\infty\,, então\; P(Y\, =1) \rightarrow\, 0 \qquad (4)$$

A utilização desse modelo para a classificação se dá pela discriminação de dois grupos, assim a regra de classificação do modelo de regressão logística é a seguinte: se *P(Y=1) > 0,5* classifica-se *Y=1*. Se *P(Y=1) < 0,5* classifica-se *Y=0*, fazendo a divisão em grupos (MIGUEL, 2020). Um exemplo de aplicação dos métodos de regressão logística é na amostra de intenções de votos, sendo possível verificar qual candidato tem mais chances de vencer considerando valores de pesquisas, investimento, dentre outros.

Segundo Mesquita (2014), embora a regressão logística incialmente fosse utilizada na área médica, a eficiência desta técnica viabilizou sua implementação em diversas áreas como: ciências médicas, estudo de mercado, intenção de voto, avaliação de crédito entre outros, tornando uma grande ferramenta de análise de dados.

Essa técnica é capaz de avaliar as probabilidades de obtenção das categorias de uma variável dependente, sendo assim, obtendo a probabilidade de ocorrência de determinado evento, assim como a influência de cada variável. Segundo Miguel (2020), o modelo de regressão logística possui as seguintes vantagens:

- Facilidade em lidar com variáveis independentes categóricas;
- Os resultados são dispostos em termos de probabilidades;
- Facilidade de classificar indivíduos em categorias;
- Neste modelo necessita de pequenos números de suposições;
- Alto grau de confiabilidade.

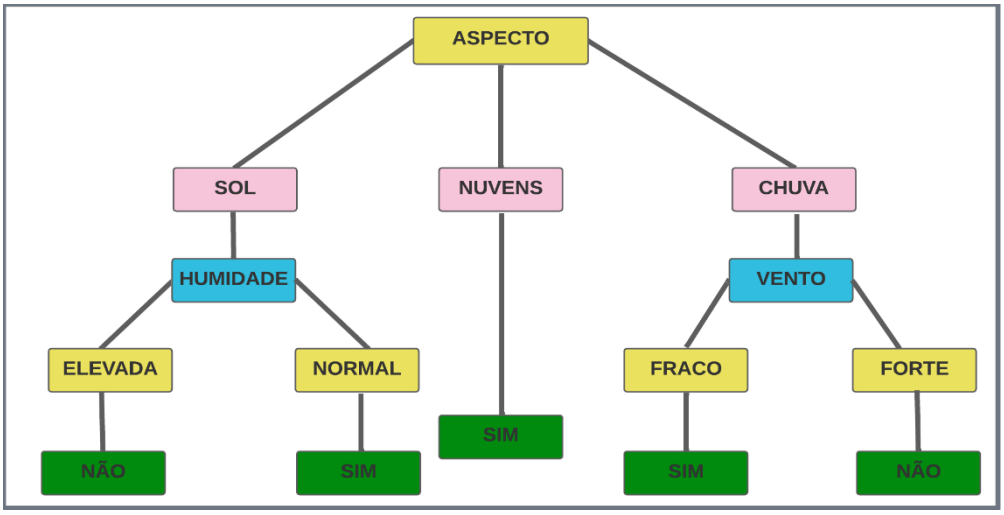
2.2.5 Árvore de decisão

Uma Árvore de decisão é uma estrutura que é composta por um fluxograma constituído por um nó raiz, nós folhas e nós internos. Esse método é atribuído regras para cada nó dentro da árvore. O nó mais alto é denominado como nó raiz, cada árvore possui apenas um único nó raiz. As ramificações dessa árvore, são denominados de

nó internos. Por fim, existem os nós folhas, também chamados de nó terminal, é considerado o final da árvore (CASTRO & FERRARI, 2016).

Essa estrutura é submetida a um algoritmo que dividirá de acordo com as possíveis classificações, sendo possível realizar outras consultadas a partir de uma característica da base de dados. Este método trata-se de uma aprendizagem de máquina supervisionado, na qual utiliza a técnica de dividir e conquistar. A Figura 8 mostra um exemplo de visualização de uma árvore de decisão para jogar bola.

Figura 8 - Visualização de uma árvore de decisão



Fonte: autoria própria.

Nesse exemplo mostrado na Figura 8, os dados possuem dois tipos de classificação final: jogar bola ou não. Para realizar essa classificação basta observar as características como o céu, a umidade e o vento.

A indução de árvores de decisão é o processo de construção de uma árvore de decisão sem objetos de classes definidas com valores do objeto. Segundo Castro & Ferrari (2016), o processo de indução se dá de maneira recursiva da seguinte forma:

- Um atributo é selecionado e colocado na raiz da árvore e são feitas ramificações de acordo com as possibilidades dos valores, sendo assim, dividindo a classe em subclasses;
- Para cada ramo que é gerado, deve-se repetir o processo de maneira recursiva;
- O final da árvore é quando todos os objetos são classificados na mesma categoria.

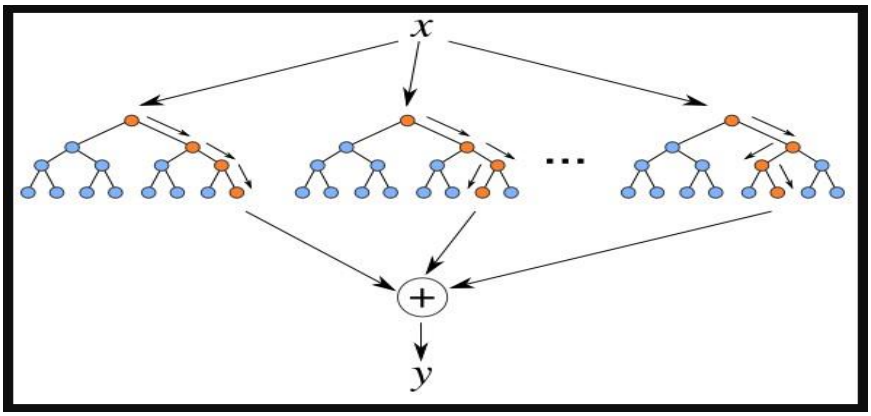
Algumas métricas de avaliações de uma árvore são dispostas, por exemplo, é necessário medir a pureza dos nós para saber qual o atributo ideal. A pureza dos nós é o quão homogêneo um nó é em relação as classes do objeto, desta forma, ao se medir a pureza dos nós é definido como será a expansão dos nós. A entropia é outra medida que define a pureza e calcula a variabilidade que pertencem ao conjunto de dados (CASTRO & FERRARI, 2016).

2.2.6 *Random Forest*

O *Random Forest (Floresta Aleatório)* é um modelo classificador de aprendizado supervisionado, este modelo deriva-se do algoritmo de árvore de decisão. O método de *random forest foi definido por* Ho (1995). Ele propôs este método devido às limitações das árvores de decisões, uma vez que, as árvores muito complexas tendem a sofrer com o fenômeno de *overfitting.* O *overfitting* é um modelo que consegue ter bons resultados no conjunto de treinamento. Entretanto, se mostra ineficiente na predição de novas entradas.

O algoritmo de *random forest* na criação de várias árvores de decisão, dentro de um vetor de testes *y = (y1, y2, y3, ..., yn)*, será classificado por cada uma dessas árvores. No fim, será avaliado através da votação (*bagging*) para decidir qual a classe mais adequada para a entrada, prevalecendo a classe mais escolhida pela maioria que compõe a floresta (HO, 1995). A Figura 9 exemplifica uma visualização do método *random forest:*

Figura 9 - Visualização do método *Random Forest*



Fonte: Baia, 2016.

A construção das árvores de decisão no método de *random forest pode ser* descrita por Cutler & Breiman (2001) da seguinte forma: considerar N vetores de entrada e cada um possui M atributos.

- Para o treinamento de uma nova árvore, é realizada uma reamostragem de tamanho N no conjunto de dados original. Ou seja, a partir da matriz primária são geradas diversas outras para treinar cada uma das árvores.
- A cada divisão de um nó, um número $m << M$ é escolhido tal que "*m*" variáveis são selecionadas aleatoriamente a partir de *M*, e a melhor divisão dessas variáveis será utilizada para dividir o nó. O valor de "*m*" será constante durante todo o treinamento
- Cada árvore cresce até o maior tamanho possível, sem que sofra nenhum tipo de poda.

O algoritmo de *random forest* apresenta vantagens como: não sofre com *overfitting;* custo computacional relativamente baixo, isto é, não demanda muito tempo para treinamentos e teste dos modelos e por fim o custo de memória limita ao armazenamento de dados.

2.2.6 Métricas de Avaliação

Segundo Neves (2022), o processo de avaliação de um conjunto de dados é realizado após o processo de treinamento do modelo, com o objetivo de verificar qual algoritmo possui a melhor acurácia. De acordo com Tan, PEI & Kumar (2011), um classificador pode ser avaliado analisando os números de exemplos de teste corretamente e incorretamente, previsto no modelo.

Uma forma de avaliação é através da matriz de confusão que possibilita uma clara visualização destes indicadores, sendo uma das ferramentas de identificação da qualidade do classificador.

Na Tabela 2 é mostrada a representação da matriz de confusão, na qual as colunas representam as classes de previsão e as linhas as classes atuais dos dados.

Tabela 2 - Matriz de Confusão

		Classe Prevista	
		0	1
Classe Correta	0	Verdadeiro Negativo	Falso Positivo
	1	Falso Negativo	Verdadeiro Positivo

Fonte: adaptado de Steinbach, Tan & Kumar, 2009.

Quando o modelo é dividido em apenas duas classes, uma é a positiva e outra é a negativa. Sendo assim, as entradas dessa matriz de confusão são definidas como:

- Verdadeiro negativo (*true negatives* – TN): representa o número de exemplos da classe negativa que está corretamente previsto como negativa;
- Falsos positivos *(false positives* – FP): são o número de exemplos da classe negativa que está previsto como positivo;
- Verdadeiro positivos (*true positives* – TP): representa o número de exemplos da classe positiva que estão corretamente previstos como positiva;
- Falsos negativos (*false negatives* – FN): são o número de exemplos da classe positiva que está previsto como negativo.

A partir da definição da matriz de confusão é possível avaliar o desempenho de classificadores como a Precisão, Taxa de Erro, Revocação, Especificidade, Exatidão e a medida *F*. A precisão pode ser definida como a porcentagem de casos corretamente classificados, taxa de erro é a porcentagem de casos incorretamente classificados.

Em classes que os resultados são majoritariamente de classes negativas, deve-se usar a medicação usando a Sensibilidade e a Especificidade. A sensibilidade, Equação (8) é a proporção de exemplos positivos que são identificados corretamente e a Especificidade, Equação (9), é a proporção de exemplos negativos que foram

identificados. Por fim tem-se a exatidão, Equação (10), que é o percentual de positivos corretamente previsto sobre o total de positivos previsto e a medida F, Equação (11) que é a média harmônica entre a exatidão e a revocação (LUNELLI, 2020).

A Tabela 3 mostra as fórmulas utilizadas nessas métricas para problemas de classificação.

Tabela 3 - Métricas para modelos de classificação

Medida	Fórmula	(5)
Acurácia	$\frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$	(6)
Taxa de erro	$\frac{\mathrm{FP} + \mathrm{FN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$	(7)
Revocação	$\frac{TP}{\mathrm{TP} + \mathrm{FN}}$	(8)
Especificidade	$\frac{TN}{\mathrm{TN} + \mathrm{FP}}$	(9)
Exatidão	$\frac{TP}{TP + FP}$	(10)
Medida *F*	$\frac{2 * exatidão * recall}{exatidão + recall}$	(11)
True Negatives	TN	(12)
False Positives	FP	(13)
True Positives	TP	(14)
False Negatives	FN	(15)

Fonte: adaptado de Lunelli, 2020.

2.2.7 Ferramentas de Mineração de Dados

O Python foi considerado a linguagem de programação mais utilizada pela comunidade de profissionais de Mineração de Dados e *Big data* (Ferreira, 2017). O Python apresenta uma sintaxe simples e objetiva, permitindo focar na resolução do problema sem se preocupar com a implementação do código. Além disso, possui uma grande comunidade, no qual, possui um vasto conjunto de bibliotecas que abrange áreas como: redes, segurança, processamento de imagens, análise e dados, entre outros. As

ferramentas e bibliotecas Python mais utilizados em *data mining* são: *Jupyter, Matplotlib,* Pandas*, Scikit-Learn.*

- *Jupyter*: baseado em protocolo cliente-servidor, permite a execução de *notebooks* via *browser. Notebooks* são documentos editáveis contendo célula de códigos e elementos visuais, assim a principal vantagem é descrição de análises e resultados de forma dinâmica e interativa (*JUPYTER*, 2022).
- *Matplotlib:* utilizada na plotagem de gráficos, muito comum na utilização de gráficos como histogramas, gráficos de barras, gráficos de pizza, entre outros (*MATPLOTLIB*, 2022).
- *Pandas:* Uma das bibliotecas mais utilizadas para análise de dados. Contém diversas ferramentas para manipulação de dados de forma simplificada (*PANDAS*, 2022).
- *Scikit*-Learn: biblioteca responsável por trabalhar com "Aprendizado de Máquinas", contém métodos de análises, processamento de dados, métricas de avaliação e algoritmos implementados (SCIKIT-LEARN, 2022).

O *Waikato Environment for Knowledge* (WEKA) é um projeto *Open Source* desenvolvido pela Universidade de Waikato na Nova Zelândia. O projeto utiliza técnicas de *Machine Learning e Data Mining, destinado aos estudos de pesquisadores*, alunos e entusiastas para resolução de problemas na área de processamento de dados, contendo algoritmos e ferramentas de mineração de dados (FRANK; et al., 2016).

O WEKA utiliza interface gráfica ou *Graphical User Interface (GUI*) e, por meio dela, é possível consultar dados de sistemas de banco de dados, executar os métodos de processamento de dados, configurar parâmetros dos algoritmos e visualizar resultados através de gráficos. É possível realizar tarefas de classificação, regressão, agrupamento e associação no software.

Sua tela inicial apresenta algumas funcionalidades, como *Explorer, Experimenter, KnowlwdgeFlow, Worbench e Simple CLI como representado na Figura 10.*

Segundo Agostini (2017), essas funções podem ser explicadas como: A opção Explorer realiza a exploração dos dados de maneira simplificada. A função *Experimenter* realiza experimentos e testes estatísticos em sistemas de aprendizagem*. KnowlwdgeFlow* é parecido com a função Explorer, contudo, seu funcionamento é através do clica e arrasta*. Workbench* apresenta todas as funcionalidades anteriores em um só lugar. E por fim, *Simple CLI* apresenta interface com linhas de comandos.

Figura 10 - Interface do Software WEKA

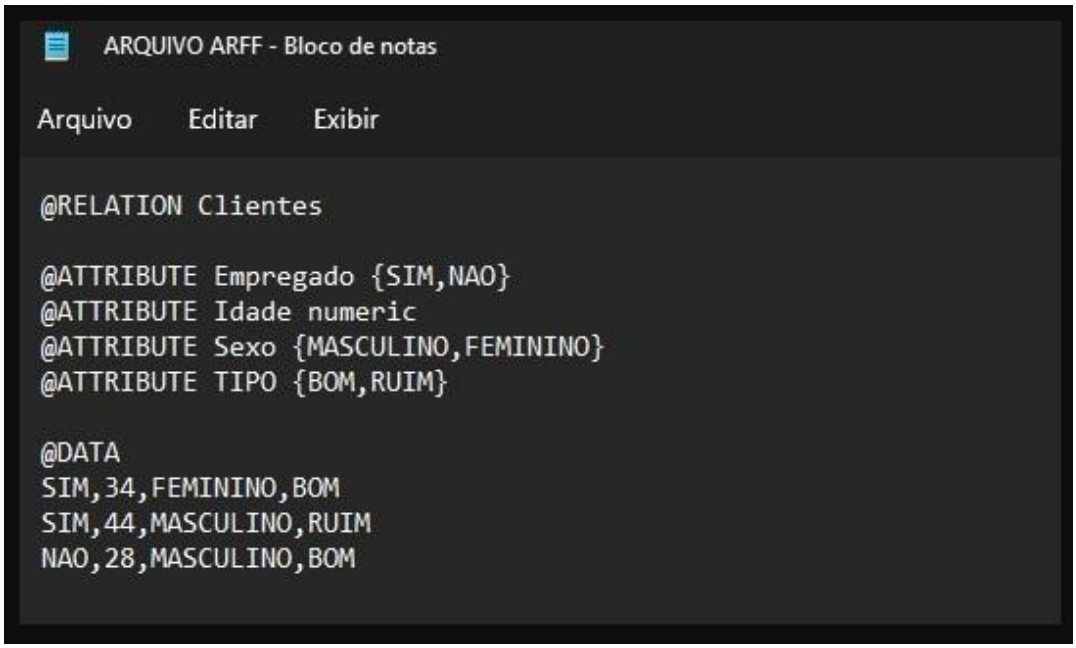
Fonte: adaptado do Software WEKA.

Para utilização de dados dentro do WEKA é necessário manipular os dados. Os dados necessitam estar em um arquivo no formato *Attribute-Relation File Format* (ARFF). Arquivos ARFF são arquivos de texto que utiliza uma estrutura de lista, em que, os registros descritos comungam dos mesmos atributos (AGOSTINI, 2017).

Um arquivo ARFF possui duas seções, em que, na primeira são declarados os atributos, é permitido a utilização do tipo *NUMERIC, DATE ou STRING.* Na segunda parte, estão os dados que são separados por vírgula, além disso, a cada nova linha é criado um registro.

Na Figura 11 é representado um arquivo básico no formato ARFF, na qual possui dados de clientes:

Figura 11 - Estrutura básica de um arquivo ARFF



Fonte: autoria própria.

O R é uma ferramenta para o desenvolvimento de sistemas de apoio a decisão e análise de dados, tal como a execução de tarefas mais complexas que envolvam programação. Uma das suas principais características é ser gratuito, estando disponível em diversas plataformas (Windows, Linux, MacOS).

Por fim, nesta seção teve como objetivo apresentar algumas das ferramentas de mineração de dados que serão utilizadas neste trabalho. Elas foram escolhidas pelo fato de serem gratuitas e de fácil utilização. Vale ressaltar, que existem diversas outras ferramentas de *Data Mining* disponíveis no mercado de ciência de dados.

2.3 Estudos correlatos

Existem trabalhos relevantes disponíveis relacionados ao uso da inteligência artificial aplicando técnicas de mineração de dados em jogos do gênero MOBAs. O trabalho de Hit-García et al. (2021) aborda a previsão da equipe vencedora em partidas profissionais do *League of Legends.* Eles utilizaram dados de pré-jogo, desta forma, são feitas análise avaliando recursos como habilidades de jogadores, sinergias entre jogadores, entre outros. Por fim, são analisados diversos algoritmos de classificação: RNN, *xgboost, svmLinear, knn, glmboost.* Dentre os algoritmos proposto o que obteve melhor precisão acima dos 70% foi o *xgbStack,* sendo compatível com as outras literaturas apresentadas no trabalho.

O trabalho de Shen (2022), aplica uma abordagem de aprendizado de máquina para prever resultados dos jogos de *League of Legends.* Ele aborda um conjunto de dados que registra os 10 primeiros minutos de partida no elo diamante. Os dados são divididos utilizando a lógica de side dentro do jogo, portanto, o conjunto de dados possui os dados do time vermelho e do time azul. Os algoritmos propostos para este trabalho foram: *AdaBoost, GradientBoost, RandomForest, ExtraTree, SVM, Naïve Bayes, KNN, LogisticRegression e DecisionTree.* Os resultados mostraram que todos os algoritmos, exceto, árvore de decisão tiveram acurácia acima dos 70%, sendo assim, a arvore de decisão apresentou apenas 63 de acurácia aos 10 minutos de partida. Por fim, são selecionados os melhores algoritmos para realizar um classificador de votação, no qual, obteve-se melhora na acurácia resultando em 72,63%.

Nascimento Jr. (2017), caracteriza perfis e comportamento de equipes no *League of Legends utilizando o modelo Cross-Industry Standard Process for Data Mining* (CRISP-DM). Os dados foram coletados do histórico de partida dos jogadores em fila ranqueada solo e ranqueadas 5x5 no servidor brasileiro, através da API da Riot Games.

Após o processo de limpeza dos dados, foi aplicado métodos estatísticos e a técnica de agrupamento através do algoritmo *K-means Clustering*, com o intuito de agrupar o desempenho das equipes e avaliar as características que influenciaram no sucesso e no fracasso das equipes através de padrões. Com o resultado concluiu que alguns grupos são mais propensos a terem equipes vitoriosas. Ao todo, foram descobertos 7 perfis de equipes que foram distribuídas em quatro níveis: Top, Bom, Mais ou Menos e Fraco.

O trabalho de Souza (2017) aplica algoritmos classificadores para previsão de vitória em uma partida de *League of Legends*. O autor dividiu o jogo e analisou o desempenho dos algoritmos em quatro momentos: momento A (de 0 a 10 minutos), momento B (10 a 20 minutos), momento C (de 20 a 30 minutos) e momento D (de 30 até o fim da partida).

A partir desta divisão, foram utilizados os algoritmos de *Random Forest* e Regressão Logística. Os dados foram coletados do trabalho de Barcellos (2017), no qual, obteve-se uma previsão média de 75% nas previsões. Segundo o autor, a variável Ouro foi o indicador de maior confiança na geração do modelo de previsões. Por fim, foi constatado que o desempenho nas fases iniciais do jogo perde importância à medida que o jogo se prolonga.

Barcellos (2017), propôs uma sistemática para avaliar quais os principais indicadores de desempenho quanto à contribuição à vitória em partidas de alto nível no *League of Legends.* O autor utilizou uma base de dados de 724.817 partidas, aplicando o método de *Generalized Estimating Equations* (GEE) e dividindo as partidas em quatro períodos. Assim, como no trabalho de Souza (2017), foi observado que os resultados iniciais iam perdendo valor conforme o jogo se prolongava.

Finalmente, Kinkade e Lim (2015), apresentaram dois tipos de preditores de vitória para o jogo DotA 2. Primeiro se baseia em dados coletados no final da partida e o segundo utiliza dados da seleção do herói. Utilizando a API da *Steam* foi possível obter uma base de dados de 62.000 partidas, na qual, utilizou-se de variáveis como Ouro/min, Experiência/min e Abates/ min. Neste trabalho foram utilizados os algoritmos de Regressão Logística e *Random Forest*, obtendo uma taxa de acerto acima de 99%. Devido à alta taxa de acerto os autores realizaram coleta de informações no início ou no meio da partida. Os resultados foram de 73% nas previsões de dados relativos à seleção dos heróis no início da partida, utilizando algoritmo de Regressão Logística, o qual concluiu ser mais eficiente do que o *Random Forest.*

3 PROCEDIMENTOS METODOLÓGICOS

Este capítulo descreve os materiais e métodos que foram utilizados para atingir o objetivo geral.

3.1 Métodos

Esta pesquisa, segundo sua natureza é um resumo de assunto. Segundo Wazlawick (2014), as pesquisas de natureza resumo de assuntos buscam explicar a área do conhecimento do projeto, indicando sua evolução histórica, como resultados da investigação das informações obtidas, levando em consideração o entendimento de suas causas e as devidas explicações.

Segundo os objetivos é uma pesquisa exploratória e descritiva. As pesquisas descritivas buscam dados consistentes sobre determinado assunto, contudo, não ocorre a interferência do pesquisador, apenas expõe os fatos como realmente são (WAZLAWICK, 2014).

A pesquisa exploratória pode ser considerada como a primeira parte do processo de pesquisa, porque não necessariamente o autor tem um objetivo ou hipótese definida (WAZLAWICK, 2014). Essa pesquisa tem como objetivo a maior familiaridade do autor com o problema, tornando intuitivo a construção de hipóteses. Geralmente é uma pesquisa flexível, porque considera os variados aspectos de fenômenos estudados (GIL, 2017).

Quanto aos procedimentos técnicos, é uma pesquisa bibliográfica e experimental. A pesquisa bibliográfica requer o estudo de teses, artigos, entre outros. A pesquisa experimental é caracterizada por ter uma ou mais variáveis experimentais que podem ser coordenadas pelo pesquisador (WAZLAWICK, 2014). Neste caso, as variáveis são manipuladas diretamente com o objeto de estudo.

3.1.1 Pesquisa Bibliográfica

A pesquisa bibliográfica, segundo Gil (2017) tem a principal vantagem em permitir uma sucessão de fenômenos maior do que seria capaz de pesquisar diretamente. Sendo assim, esta pesquisa será elaborada a partir de materiais já publicados, podendo incluir livros, teses, materiais disponibilizados na Internet, revistas, entre outros.

De acordo com Gil (2017), as características da pesquisa bibliográfica se desenvolvem a partir das seguintes etapas:

a) A escolha de um tema: Utilização de Data Mining no *League of Legends;*
b) Levantamento bibliográfico preliminar: Foi realizado levantamento bibliográfico sobre os principais trabalhos sobre *Data Mining* e os principais autores que são referenciados nos artigos.
c) Formulação do problema: **Quais fatores influenciam na classificação dos resultados de uma partida baseado no desempenho das equipes no contexto do jogo *League of Legends*, em dois momentos distintos do jogo?**
d) Busca das fontes: A partir disso, foi identificado fontes bibliográficas de maior valor, capazes de fornecer informações para ser possível responder o problema proposto, consultando dissertações, obras de referência, entre outros.
e) Leitura do material: Realizou-se leituras do material adquirido estabelecendo relações com o problema proposto e analisar a coerência das informações e dados apresentados pelos autores.
f) Fichamento: Foi anotado as ideias que surgiram, identificando as informações relevantes e registrando os comentários das obras.
g) Redação do texto: Escrita final do trabalho de conclusão realizada para o TCC II.

3.1.2 Pesquisa Experimental

A pesquisa experimental, segundo (WAZLAWICK, 2014) consiste que o pesquisador provoque mudanças no ambiente de pesquisa, observando as alterações realizadas são de acordo com os resultados esperados pela pesquisa.

Dentre as características da pesquisa experimental são: estabelecer um objeto de estudo, escolher as variáveis que a influenciam e determinar as formas de controle e observar os efeitos que a variável gera no objeto. Realiza pelo menos um dos elementos que julga ser responsável pela circunstância que está sendo pesquisado (GIL, 2017).

Segundo Gil (2017), a pesquisa experimental pode ser composta das seguintes etapas:

a) Formulação do problema: **Quais fatores influenciam na classificação dos resultados de uma partida baseado no desempenho das equipes no contexto do jogo *League of Legends*, em dois momentos distintos do jogo?**

b) Definição do plano experimental: este trabalho foi divido em cinco etapas para sua realização:

A primeira etapa deste trabalho refere-se à coleta de dados do histórico de partida das principais ligas profissionais de *League of Legends* no ano de 2022*: League of Legends Championship Series da América do Norte (*LCS), *League of Legends Championship Series da Europa (*LEC), *League of Legends Champions Korea* (LCK), *League of Legends Pro League China (*LPL), *Pacific Championship Series (*PSC), Campeonato Brasileiro de *League of Legends* (CBLoL) entre outras. Os dados são coletados através site *Oracle's Elixir*, no qual, refere-se a sites de categoria estatístico que retira dados através da API *Riot Games.* A partir disso, ocorreu a remoção de variáveis que não condiziam com o desempenho de equipes, além de variáveis redundantes e variáveis não desejadas para observação.

A segunda etapa é definida em duas etapas, a primeira é a criação de um objeto de estudo, nesta etapa ocorreu a criação de dois modelos a serem analisados, o primeiro analisa os 15 primeiros minutos de partida e o segundo modelo representa os dados dos jogos ao final de partida. A segunda etapa é a preparação dos dados, nesta etapa ocorreu a limpeza dos dados, adequação dos dados, *feature selection* pelo método *SelectKBest* das principais variáveis e análise de variáveis altamente correlacionadas. Desta forma, foi utilizado o Jupyter Notebook juntamente com as principais bibliotecas de análise de dados do *Python*, além disso, foi utilizado dados no excel em formato *Comma-separated Values (CSV)* para uma melhor visualização, por fim, foi aplicado técnicas de mineração de dados utilizando o python como ferramenta de mineração.

A terceira etapa representa a aplicação dos algoritmos de Regressão Logística, *Random Forest* e Árvore de Decisão utilizando o WEKA para ambos os modelos. As fases de teste e treinamentos foi adotado a utilização da técnica de 10-*fold cross-validation* (validação cruzada) disponibilizada no WEKA.

Na quarta etapa foi analisado os resultados referentes à aplicabilidade dos algoritmos de *machine learning*, mostrando qual teve o melhor desempenho para cada modelo no contexto do jogo *League of Legends*.

Finalmente, a quinta e última etapa refere-se à escrita dos resultados na pesquisa e as conclusões finais sobre o trabalho.

c) Determinação do ambiente: Na fase de preparação dos dados foi utilizado o *Jupyter Notebook* na versão 6.4.12*,* utilizando as bibliotecas como *pandas* versão 1.4.4*, matplotlib* versão 3.5.2*, seaborn* versão 0.11.2*, numpy* versão 1.21.5, *scikit-learn* versão 1.0.2 do Python versão 3.9.13. Além disso, a fase de experimento dos algoritmos de *machine learning* foi utilizado o software WEKA, na versão 3.8.6.

Foi utilizado dois tipos de equipamentos para o desenvolvimento deste trabalho: um computador com o sistema operacional (SO) Windows, processador *Intel Core* i5-10400F com 2.90 *GigaHertz* (GHz), placa de vídeo *Ray Tracing Extreme (*RTX) 3070 8 *GigaByte* (GB), 16 GB de Memória de Acesso Randômico (RAM) e armazenamento de 1 *TeraByte* (TB) *Solid State Drive* (SSD).

O outro equipamento é o notebook *Lenovo ideapead Gaming 3i* com o sistema operacional Windows, processador Intel Core i5-10300H com 2.50 GHz, placa de vídeo de vídeo *Giga Texel Shader eXtreme* (GTX) 1650 4GB, 24 GB de memória RAM e armazenamento de 256 GB SDD.

d) Coleta de dados: Os dados foram coletados através do site *Oracle's Elixi*r em formato .csv, no qual, contém dados do histórico de partida das principais competições de *League of Legends* no período de 2022. Desta forma, selecionando as variáveis que condiz com os dois modelos de estudos, excluídos variáveis redundantes e dados faltantes.

e) Análise e interpretação dos dados: Os dados de análises foram obtidos através dos testes utilizando o Python. Com os resultados obtidos pelo Python foram realizadas a interpretação dos resultados, verificando qual algoritmo atendeu melhor cada modelo proposto.

f) Redação do relatório: Escrita do TCC II.

4 PREPARAÇÃO E DESCOBERTA DE CONHECIMENTO EM BASE DE DADOS

Este capítulo descreve os processos de preparação dos dados utilizando o processo de descoberta de conhecimento de bases de dados (KDD) para os modelos propostos. Todo o processo de preparação de dados aconteceu através do *Juyper Notebook* versão 6.4.12, utilizando as bibliotecas *pandas* versão 1.4.4*, matplotlib* versão 3.5.2*, seaborn* versão 0.11.2*, numpy* versão 1.21.5, *scikit-learn* versão 1.0.2 do *python* versão 3.9.13.

O primeiro modelo analisa os resultados das partidas nos 15 primeiros minutos de partida, sendo selecionadas as variáveis que possuem relação com esse tempo de jogo. O segundo modelo analisa os resultados das partidas, usando todas as variáveis coletadas ao final do jogo. Vale ressaltar, que foram utilizadas as mesmas partidas para analisar ambos os modelos, sendo possível através do *dataset*, contudo, para cada modelo possui variáveis específicas.

Inicialmente foram selecionadas as variáveis através do processo de seleção dos dados, modelagem das características dos dados, criação de um objeto de estudo. Essa fase inicial teve como objetivo remover variáveis que não tem relação com o desempenho de equipes, variáveis redundantes e adequação dos dados.

Também é exemplificado quais foram as variáveis selecionadas para os modelos, aplicado todo o processo de limpeza dos dados, remoção de *outliers* e *SelectKBest* que é um método de *Feature Selection* para selecionar as variáveis principais. Além disso, foi aplicado o cálculo de métricas, transformação dos dados para escala de min-max e análise de correlação para remover as variáveis altamente correlacionadas.

4.1 Seleção dos Dados

A *Riot Games*, desenvolvedora do *League of Legends*, fornece uma API baseada na Web, na qual é possível acessar dados dos históricos das partidas em formato *JavaScript Object Notation* (JSON). Diversos sites utilizam desses dados para criar produtos como sites de estáticas de histórico de partida, acompanhar partidas em tempo real, rank dos melhores jogadores de cada região, entre outros. Um dos sites mais famosos nesses segmentos é o *League of Graphs e o OP.GG, ainda sim,* para utilizar

os conteúdos fornecidos pela API é necessário registrar o produto no portal *Riot Developer* e respeitar os termos e políticas da *Riot Games*.

O processo de coleta de dados foi realizado através do site *Oracle's Elixir*, no qual, é disponibilizado o conjunto de dados através do Google Drive em arquivo CSV. O site tem foco na coleta de dados dos principais campeonatos de *League of Legends* diariamente como *League of Legends Championship Series da América do Norte (*LCS), *League of Legends Championship Series da Europa (*LEC), *League of Legends Champions Korea* (LCK), *League of Legends Pro League China (*LPL), *Pacific Championship Series (*PSC), Campeonato Brasileiro de *League of Legends* (CBLoL) entre outras.

Todos os jogos de campeonatos oficiais são referentes ao modo clássico de jogo, sendo cinco jogadores de cada lado em duas equipes no mapa *Summoner's Rift.* Para este trabalho foi realizado o *download* da base de dados do ano de 2022 contendo o histórico de partida das principais ligas. O conjunto de dados possui um total de 149.400 registros e um total de 123 variáveis.

Através do *gameid* (chave primária da criação de uma partida) os dados são divididos entre os cinco primeiros registros de jogadores individuais da equipe A. Posteriormente, os cinco registros dos jogadores da equipe B e, por fim, duas linhas que é o somatório do desempenho das equipes A e B. Desta forma, cada histórico de partida aloca um total de 12 linhas no conjunto de dados.

A utilização de uma base de dados previamente coletada efetuou-se por ser um método mais prático para o objetivo do trabalho. Quando se fala em predizer partidas no âmbito do *League of Legends,* logo vem na mente a geração de conhecimentos na área de apostas esportivas voltadas aos principais campeonatos. A coleta de dados ocorre normalmente através da variável *gameid, sendo* possível coletar dados selecionando valores randômicos ou através de uma lista de *gameid* que se deseja coletar.

Através disso, foi observado que para fazer a coleta de dados das principais competições, deveriam ter algum filtro que fosse capaz de selecionar apenas essas partidas. Entretanto, é necessário que tenha uma lista de *gameid* das principais competições, aumentando consideravelmente a dificuldade do trabalho.

4.1.1 Modelo de Características dos Dados

As características do modelo de dados relacionado ao desempenho de equipes podem ser representadas:

Seja um conjunto de históricos de partidas *M*, um conjunto de equipes *T* e um conjunto dimensional de desempenho de jogadores representado por *P*. Um histórico de partida *m € M* pode ser representado por duas equipes distintas *m,* cada equipe *t € T* consiste no desempenho sumarizado de 5 jogadores da mesma equipe, e cada desempenho de jogador *p € P* pode ser representado como um vetor de estatísticas de um jogador individualmente, tais como *kills, deaths, assists, firstblood,* entre outras.

Após Coletado um conjunto de dados *M* = 149.400 registros de históricos de partida. As 12 linhas que possuem o mesmo *gameid* neste conjunto de dados representa os dados de jogadores individuais e dados sumarizados das equipes. Foi feita uma análise que corresponde a *T* = 24.900 equipes e *P* = 124.500 desempenho de jogadores individuais, totalizando o valor de *M*. Por fim, dividindo o valor de *T* por dois é possível identificar que o conjunto de dados possui um total de 12.450 partidas registradas.

Após feita a primeira análise no conjunto de dados foi observada que por conta de já existir dados sumarizados relacionados ao desempenho de equipes foi aplicado um filtro removendo os dados de *P* (124.500 dados relacionados ao desempenho dos jogadores individualizados) do conjunto de dados. Caso contrário, se não existissem esses dados, a fim de analisar os dados de uma equipe deveria ser feito uma sumarização dos dados para obter o desempenho de equipes.

Por fim, analisando a base de dados através da variável chamada *datacompleteness* que identifica os registros que estão completos e registros parciais. Isso acontece por conta de alguns registros não possuir dados em todas as 123 variáveis, desta forma, foi aplicado um filtro para extrair somente os registros que estão completos.

Em síntese, a nova base de dados de desempenho de equipes representa por X possui um total de 21.174 registros distribuído em 10.587 partidas. Modelando o conjunto *X*, pode-se representar uma matriz de dimensões 21.174 x 123.

4.2 Criando um Objeto de Estudo

A partir de uma base de dados com alta dimensionalidade é importante fazer uma seleção cuidadosa das variáveis. Muitas das vezes, em uma análise dos dados nem

todas as variáveis são igualmente relevantes. Por exemplo, os conjuntos de dados contêm informações redundantes, dados incompletas ou irrelevantes, levando para conclusões imprecisas ou enganosas.

O conjunto de dados trabalhado possui variáveis do histórico de partida de todos os tipos, ao todo são 123 variáveis. Algumas dessas variáveis não contém relação nenhuma com o desempenho de equipes nas partidas do jogo *League of Legends*.

Com o intuito de diminuir a dimensionalidade dos dados e obter um objeto de estudo, este capítulo descreve os processos de análise de variáveis, através da observação, análise de variáveis redundantes, variáveis não desejadas e a adequação dos dados.

4.2.1 Análise das Variáveis por Observação

Inicialmente foi feita uma análise verificando quais variáveis não têm relação direta com o desempenho de um time. Os históricos de partida contêm dados como quantidade de abates, quantidades de mortes, quantidade de assistência, ouro gasto, dano total causado a campeões inimigos, entre outros. Desta forma, foi removido variáveis que não tem relação com o desempenho do time e variáveis que não possui dados, por exemplo, variáveis exclusivamente de desempenho de jogadores. O Quadro 1 exemplifica as variáveis que foram removidas do *dataset* e sua descrição:

Quadro 1 - Variáveis removidas por observação

Variáveis	**Descrição**
gameid	Identificador de criação de partida (removida após todo o processo de adequação dos dados)
gameleng	Duração da partida (removida após a transformação dos dados)
datacompleteness	Indicador de dados de partidas completas ou parciais.
url	Link dos dados dos jogos
league	Nome da liga
year	Ano de criação do game

split	Divisão de jogo (Ex: Spring, Summer)
playoffs	Indicador de jogos decisivos
date	Data do game
patch	Versão do Patch do jogo
participantid	Identificador do jogador no jogo
side	Posição que o time se encontra, lado azul do mapa ou lado vermelho (Removida após a adequação dos dados)
position	Posição do jogador (Top, Jungle, Mid, ADC ou Support)
playername	Nome do jogador
playerid	Identificador do jogador
teamname	Nome do time
champion	Nome do Campeão escolhido pelo jogador
ban1	1º campeão banido da equipe
ban2	2º campeão banido da equipe
ban3	3º campeão banido da equipe
ban4	4º campeão banido da equipe
ban5	5º campeão banido da equipe
firtsbloodkill	Jogador da partida que fez a primeira eliminação no jogo
firstbloodassist	Jogador que deu assistência para a primeira eliminação no jogo
firstbloodvictim	Jogador que morreu na primeira eliminação
dragons (type unknown)	Dragões (Tipo Desconhecido)
damageshare	Jogador que compartilhou o dano

earnedgoldshare	Jogador que ganhou parte do ouro
total cs	Quantidade de minions e monstros neutros que o jogador derrotou
monsterkillsownjungle	Monstro que o jogador matou da própria selva
monsterkillsenemyjungle	Monstro que o jogador matou da selva inimiga

Fonte: autoria própria

A partir do Quadro 1 fica evidente as variáveis retiradas do conjunto de dados através da observação das variáveis que não tem relação nenhuma com o desempenho das equipes. Vale ressaltar, que variáveis como *gameid* (Identificador do game), *gameleng* (duração da partida) e *side* (lado do mapa) foram previamente mantidas na base de dados para realização da adequação, limpeza e transformação dos dados.

Ainda assim, analisando o conjunto de dados foi observado que existiam variáveis exclusivamente de jogadores individuais que não possuíam dados, isto é, conjunto de dados vazios. Após verificação, as seguintes variáveis *firtsbloodkill, firstbloodassist, firstbloodvictim, dragons (type unknown), damageshare, earnedgoldshare, total cs, monsterkillsownjungle e monsterkillsenemyjungle* foram removidas do *dataset* conforme mostrado no Quadro 4. Ao todo, foram removidas 30 variáveis nesta etapa, restando 93 variáveis do conjunto de dados original.

4.2.2 Análise de Variáveis Redundantes e Variáveis não Desejadas

Em seguida, a partir de uma análise minuciosa foi observado variáveis que tem relação com o desempenho das equipes no jogo *League of Legends,* contudo, essas variáveis fornecem informações semelhantes ou repetidas, ou seja, variáveis redundantes.

Um dos grandes problemas de trabalhar com variáveis redundante na mineração de dados é diminuir a eficiência do modelo, produzir resultados impreciso e aumento no tempo de processamento. Além disso, variáveis redundantes são altamente correlacionadas entre si, tornando difícil analisar qual variável contribui mais em um modelo preditivo.

O conjunto de dados possui diversas variáveis redundantes, por exemplo, as variáveis *dragons e opp_dragons.* A primeira variável representa a quantidade de

dragões que foram abatidos pelo time aliado e a segunda variável representa a quantidade de dragões abatida pelo time inimigo.

Seja uma partida, na qual a primeira equipe matou um total de 2 dragões e a segunda equipe matou um total de 4 dragões. Os dados do time A seriam *dragons = 2* e opp_dragons = *4,* da mesma forma, os dados da equipe B seriam *dragons = 4 e opp_dragons = 2* gerando redundância dos dados. Para melhorar a qualidade dos resultados da mineração de dados foi definido a remoção dessas variáveis redundantes representadas por dados dos oponentes, ou seja, variáveis como *opp_dragons, opp_elementaldrakes,* dentre outras.

Outro tipo de variável redundante encontrada está relacionado as variáveis que são representadas tanto por seus valores brutos como transformada para o tempo de jogo em minutos. Visto que não são todas as variáveis que são transformadas em escala de tempo, em um primeiro momento foi removido todas as variáveis que têm essa característica, mantendo apenas as variáveis de valores brutos. Posteriormente, no processo de transformação dos dados, todas as variáveis foram normalizadas, descrita na seção 4.6.1, através do cálculo de métricas.

Algumas variáveis não foram selecionadas para o propósito do trabalho, sendo intituladas como variáveis não desejadas. Por exemplo, as variáveis *infernals, mountain, clouds, oceans, chemetech e hextechs.* Essas são variáveis que mostram a quantidade de tipos de dragões que um time abateu durante a partida.

Em síntese, tipo de variável pode aumentar bastante a dimensionalidade dos dados, isto é, possuir grande dimensionalidade, pode afetar o desempenho dos modelos de *machine learning.* A partir dessa análise, foram selecionadas algumas variáveis não desejadas e feita a remoção no *dataset*.

Além disso, existem variáveis que referenciam o tempo de jogo aos 10 minutos, por exemplo, *goldat10, xpat10, golddiffat10, killsat10, entre outras.* Por conta de já existir um modelo dos 15 primeiros minutos de partida foi desconsiderado analisar os 10 minutos de partida, pois ambos estão no *early game* com pouca diferença entre os tempos*,* isto é, os dados obtidos estão relacionados à fase inicial de jogo.

O Apêndice A mostra o Quadro 4 com as variáveis que foram removidas por redundância e as variáveis não desejadas para o propósito do trabalho indicada com (*) removidas do *dataset.* Nesta segunda etapa foram removidas um total de 49 variáveis, sendo 21 variáveis redundantes e 27 variáveis não desejadas.

4.2.3 Adequação dos Dados

Tendo em vista que os dados no *dataset* ocupam duas linhas, sendo uma para a equipe A e outra para a equipe B, é necessário fazer a junção desses registros para que os dados de ambas as equipes fiquem no mesmo dado. A Figura 12 exemplifica os registros sem a adequação do modelo 2.

Figura 12 – Registros sem adequação

gameid	side	gamelength	result	kills	deaths	assists	doublekills	triplekills	quadrakills	...	firstmidtower	firsttothreetowers	inhibitors	damagetochampions
2690210	Blue	1713	0	9	19	19	0.0	0.0	0.0	...	1.0	1.0	0.0	56560.0
2690210	Red	1713	1	19	9	62	6.0	0.0	0.0	...	0.0	0.0	1.0	79912.0
2690219	Blue	2114	0	3	16	7	0.0	0.0	0.0	...	0.0	0.0	0.0	59579.0
2690219	Red	2114	1	16	3	39	1.0	0.0	0.0	...	1.0	1.0	2.0	74855.0

Fonte: autoria própria.

Para realizar a adequação dos dados, foi utilizada a lógica de *Side*. A *Side* refere às posições dos times no mapa, durante uma partida. Os lados são divididos entre *Blue Side,* que nada mais é do que o time que está do lado azul do mapa e o *Red Side* que é o time que está do lado vermelho do mapa. A Figura 13 mostra como é interpretado a side, dentro do jogo *League of Legends.*

Figura 13 – Mapa de *Summoner's Rift* visualizada pela Side

Fonte: Hitar-García et al., 2021.

Através da variável *gameid* e da variável *side* foram colocados prefixo em todas as variáveis tanto do modelo 1 como no modelo 2. As variáveis que têm valores binários, foram como *Blue_firstblood, Red_firstblood, Blue_firsttower, Red_firsttower*, além da variável *target* Blue_result e Red_result, foram mantidas apenas a variável com prefixo *blue*.

Desta forma, essas variáveis quando recebem o valor 1 está referenciando à equipe azul e, quando recebem o valor 0, está referenciando o time vermelho. A Figura 14 mostra os dados do modelo 2, após a adequação dos dados.

Figura 14 - Registros adequados

gameid	side	Blue_gamelength	Blue_result	Blue_kills	Blue_deaths	Blue_assists	Blue_doublekills	Blue_triplekills	Blue_quadrakills	...	Red_firstmidtower
2690210	Blue	1713	0	9	19	19	0.0	0.0	0.0	...	0.0
2690219	Blue	2114	0	3	16	7	0.0	0.0	0.0	...	1.0
2690227	Blue	1972	1	14	5	42	3.0	1.0	0.0	...	0.0
2690255	Blue	2488	0	16	13	41	3.0	0.0	0.0	...	0.0
2690264	Blue	2020	1	13	8	37	1.0	0.0	0.0	...	0.0

Fonte: autoria própria.

Após feita a adequação dos dados os registros caíram de 21.174 para 10.587, em ambos os modelos. Além disso, no modelo 1 teve um total de 24 variáveis selecionadas e no modelo 2 um total de 47 variáveis para o objeto de estudo.

4.3 Variáveis que compõem os modelos

A partir da criação de um objeto de estudo de extrema importância descrever e entender cada variável que foi selecionada para este trabalho. Desta forma, essa seção exemplifica as variáveis selecionadas para os dois modelos propostos.

4.3.1 Modelo 1 – Desempenho das equipes aos 15 minutos de partida

O Quadro 2 mostra a descrição de cada uma das 24 variáveis selecionadas de medição do desempenho de uma equipe aos 15 minutos de partida, no contexto do *League of Legends:*

Quadro 2 – Variáveis de desempenho das equipes aos 15 minutos de partida

Variáveis	**Descrição**
gameleng	Duração da partida em minutos.
Blue_result	Variável *target.* Recebe 1 quando a vitória é do time azul e 0 quando a vitória é time vermelho do mapa.
Blue_firstblood	Primeira kill do jogo, recebe 1 quando a kill foi para o lado azul e 0 quando a kill foi para o lado vermelho.
Blue_firstdragon	Primeiro dragão do jogo, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_firstherald	Primeiro arauto do jogo, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_firsttower	Primeira torre do jogo, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_goldat15 *Red_goldat15*	Quantidade de ouro recebido pelo time Blue e pelo time Red aos 15 minutos de partida.
Blue_xpat15 *Red_xpat15*	Quantidade de experiência obtida pelo time Blue e pelo time Red aos 15 minutos de partida.
Blue_csat15 *Red_csat15*	Quantidade de tropas abatidas pelo time Blue e pelo time Red aos 15 minutos de partida.
Blue_golddiffat15 *Red_golddiffat15*	Diferença de ouro entre os times aos 15 minutos de partida.
Blue_xpdiffat15 *Red_xpdiffat15*	Diferença de experiência entre os times aos 15 minutos de partida.
Blue_csdiffat15 *Red_csdiffat15*	Diferença de tropas abatidas entre os times aos 15 minutos de partida.
Blue_killsat15 *Red_killsat15*	Quantidade de abates conseguidas pelo time Blue e pelo time Red aos 15 minutos de partida.
Blue_assistsat15 *Red_assistsat15*	Quantidade de assistência conseguidas pelo time Blue e pelo time Red aos 15 minutos de partida.
Blue_deathsat15 *Red_deathsat15*	Quantidade de mortes cedidas pelo time Blue e pelo time Red aos 15 minutos de partida.

Fonte: autoria própria.

Diante do exposto, vale ressaltar que existem variáveis que não podem ser selecionadas para o modelo preditivo de análise aos 15 minutos de partida. Por exemplo, as variáveis *first_baron, Blue_barons e Red_barons* refere-se ao objetivo neutro baron nashor, que aparece na partida somente aos 20 minutos de partida.

Outro tipo de variável não selecionada relaciona às varáveis *Blue_inhibitors Red_inhibitors.* Quando ocorre a destruição dos inibidores significa que a partida está muito próxima de acabar. Sendo assim, para que não ocorra o desbalanceamento do modelo, essas variáveis de final de jogo não foram selecionadas.

4.3.2 Modelo 2 – Desempenho das equipes ao final do jogo

O Quadro 3 exibe a descrição de cada uma das 47 variáveis selecionadas de medição do desempenho de uma equipe ao final da partida de uma partida de *League of Legends:*

Quadro 3 – Variáveis do desempenho das equipes ao final do jogo

Variáveis	**Descrição**
gameleng	Duração da partida em minutos.
Blue_result	Variável *target.* Recebe 1 quando a vitória é do time azul e 0 quando a vitória é time vermelho do mapa.
Blue_firstblood	Primeira kill do jogo, recebe 1 quando a kill foi para o lado azul e 0 quando a kill foi para o lado vermelho.
Blue_firstdragon	Primeiro dragão do jogo, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_firstherald	Primeiro arauto do jogo, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_firstbaron	Primeiro barão do jogo, recebe 1 quando for para o lado azul e 0 quando for do lado vermelho.
Blue_firsttower	Primeira torre do jogo, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_firstmidtower	Primeira torre do mid destruída, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_firstthreetowers	Primeiras três torres destruídas do jogo, recebe 1 quando para o lado azul e 0 para o lado vermelho.
Blue_kills *Red_kills*	Quantidade de abates conseguidas pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_deaths *Red_deaths*	Quantidade de mortes cedidas pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_assists *Red_assists*	Quantidade de assistência conseguidas pelo time Blue e pelo time Red ao longo de todo o jogo.

Blue_doublekills Red_doublekills	Quantidade de dois abates seguidos que uma equipe conseguiu ao longo de todo o jogo.
Blue_triplekills Red_triplekills	Quantidade de três abates seguidos que uma equipe conseguiu ao longo de todo o jogo.
Blue_quadrakills Red_quadrakills	Quantidade de quatros abates seguidos que uma equipe conseguiu ao longo de todo o jogo.
Blue_pentakills Red_pentakills	Quantidade de cinco abates seguidos que uma equipe conseguiu ao longo de todo o jogo.
Blue_dragons Red_dragons	Números de dragons abatidas pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_heralds *Red_heralds*	Números de arautos abatidos pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_barons *Red_barons*	Números de baron nashor abatidos pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_towers *Red_towers*	Números de torres destruídas pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_inhibitors Red_inhibitors	Quantidade de inibidores destruídos pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_damagetochampions *Red_damagetochampions*	Quantidade de danos causados aos inimigos pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_wardsplaced *Red_wardsplaced*	Quantidade de sentinelas de controle colocadas pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_wardskilled Red_wardskilled	Quantidade de sentinelas de controle destruída pelo time Blue e pelo time Red ao longo de todo o jogo.
Blue_visionscore e *Red_visionscore*	Pontuação de visão adquirida ao logo do jogo.
Blue_totalgold Red_totalgold	Quantidade de ouro recebido (ouro ganho, ouro de recompensas, ouro de objetivos).
Blue_earnedgold Red_earnedgold	Quantidade de ouro obtido exclusivamente pela equipe.
Blue_cspm *Red_cspm*	Quantidade média de tropas inimigas abatidas por minuto de partida.

Fonte: autoria própria.

Para uma melhor separação dos objetos de estudo, neste modelo 2 não foi selecionada nenhuma variável que tenha relação com o tempo de jogo, por exemplo, *goldat10, killsat10, goldat15, killsat15,* entre outras. Desta forma, apenas as variáveis de final de jogo foram selecionadas para este modelo.

4.4 *SelectKBest*

Em geral, tarefas de pré-processamento envolve a seleção de variáveis mais relevantes, também chamada de *Feature Selection* ou Seleção de Recursos. O objetivo de utilizar o *Feature* é justamente reduzir a dimensionalidade do conjunto de dados, removendo variáveis irrelevantes, redundantes etc.

Diante disso, vale ressaltar que existem diversas técnicas que utilizam, por exemplo, técnicas baseadas em filtros que utilizam métricas estáticas. Para realizar a escolha das melhores variáveis para o modelo preditivo, de ambos os modelos, foi escolhido o método *SelectKBest*.

O *SelectKBest* é um método de transformação, no qual, remove todos, exceto os atributos de maior pontuação. Esse método utiliza a *Univariate feature selection* ou seleção univariada de atributos), selecionando os melhores atributos com base em testes estatísticos univariados (SCIKIT-LEARN DEVELOPERS, 2023).

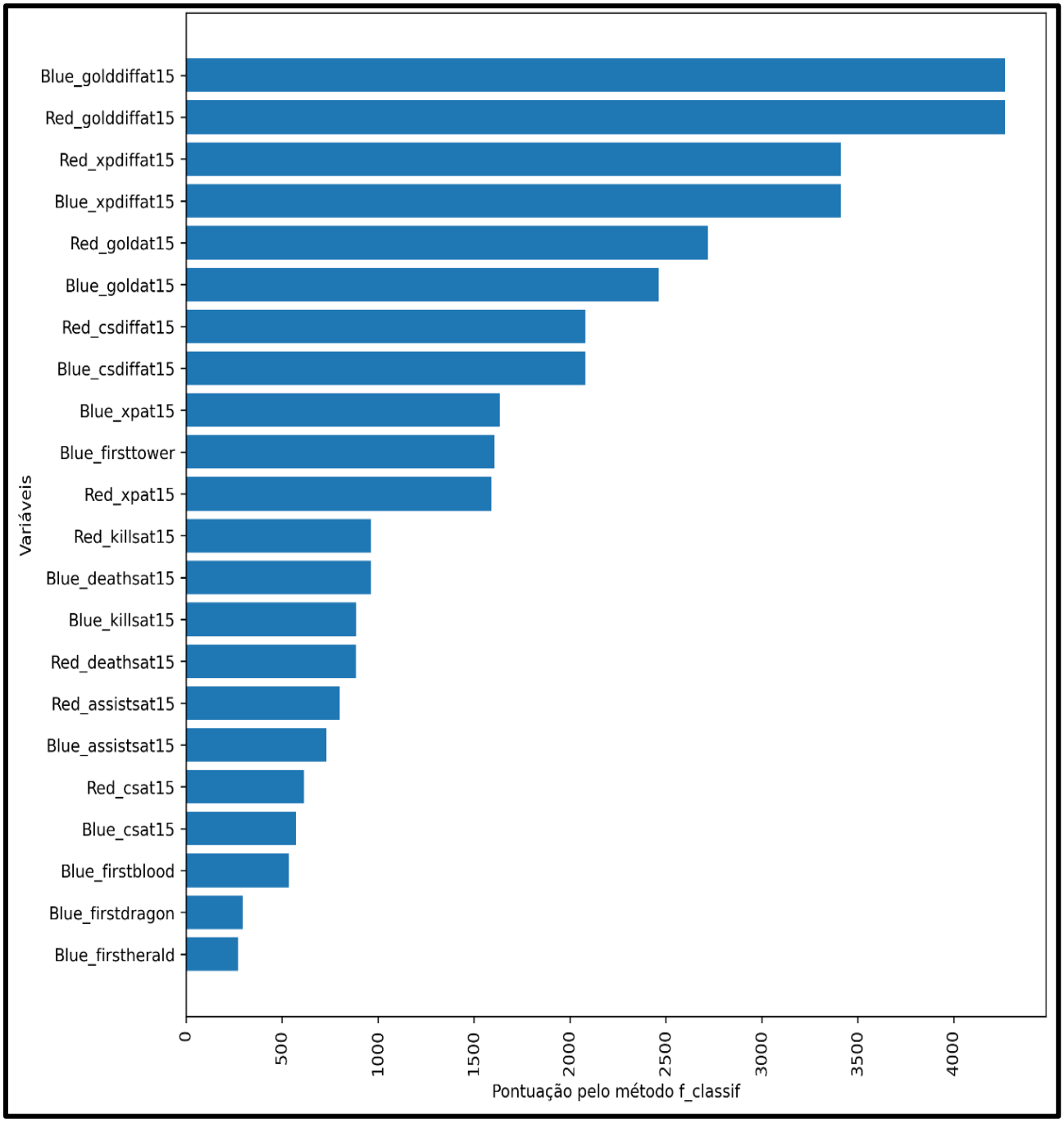
Este método está intercalado à biblioteca *sklearn.feature_selection.SelectKBest* do *python*, sendo possível definir o valor de K melhores variáveis do modelo, contudo, não foi definido um valor específico para K, desta forma, é possível observar a pontuação de todas as variáveis dos modelos. O método possui diversas funções como *f_classif, chi2, f_regression, SelectPercentile,* entre outras.

Para este trabalho foi selecionado o método *f_classif* para problemas de classificação. Além disso, é necessário definir a variável *target Blue_result* para a classificação. A vantagem de utilizar esse método se dá pelo fato de relacionar o grau de importância de todas as variáveis em relação à variável *target.*

A Figura 15 e 16 mostram um gráfico de barras com as melhores variáveis para os dois modelos, sendo assim, é possível visualizar a importância de todas as variáveis nos modelos.

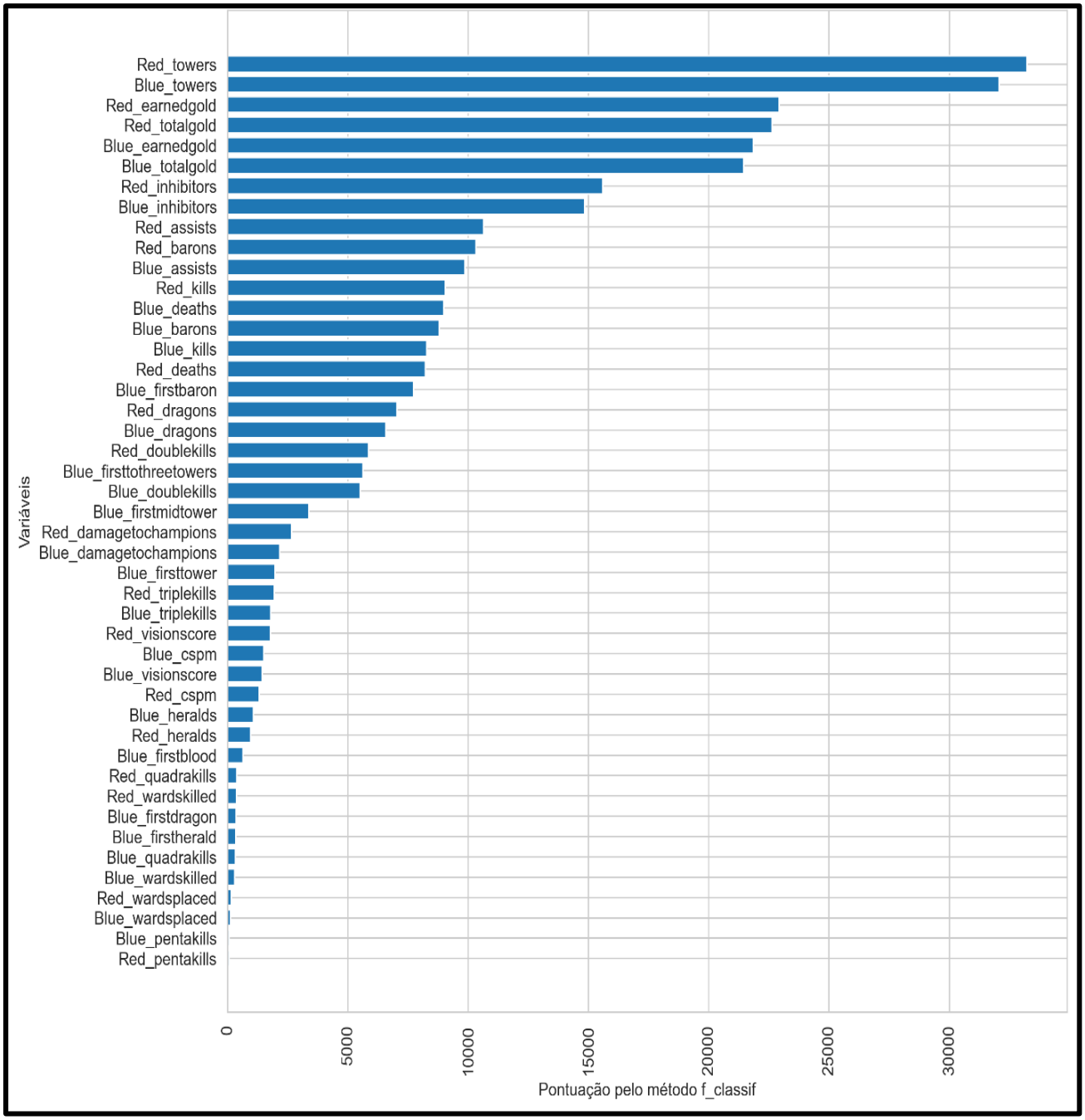
De acordo com as Figuras 15 e 16 ambos os modelos possuem variáveis com baixa pontuação, sendo assim, essas variáveis com baixa pontuação foram removidas da análise de dados. Para o primeiro modelo as variáveis Red_assistsat15, Blue_assistsat15, Red_csat15, Blue_csat15, Blue_firstblood, Blue_firstdragon e Blue_firstherald foram removidas. Ao todo, foram removidas um total de 7 variáveis, restando 17 variáveis no primeiro modelo.

Figura 15 - *Feature Selection SelectKBest* do primeiro modelo.



Fonte: autoria própria.

Figura 16 - *Feature Selection SelectKBest* do segundo modelo



Fonte: autoria própria.

O segundo modelo teve um maior número de variáveis com baixa pontuação em relação à variável *target*. Desta forma, as seguintes variáveis foram removidas: *Red_wardsplaced, Blue_wardsplaced, Blue_pentakills, Red_pentakills, Red_wardskilled, Blue_wardskilled, Blue_firstherald, Blue_visionscore, Blue_firstdragon, Blue_quadrakills, Red_quadrakills, Blue_firstblood, Red_visionscore, Blue_damagetochampions, Blue_heralds, Red_heralds, Red_damagetochampions,*

Red_cspm, Blue_cspm, Blue_firsttower, Blue_triplekills, Red_triplekills e Blue_firstmidtower.

É importante destacar que algumas variáveis como Blue_pentakills, Red_pentakills, Blue_quadrakills e Red_quadrakills era esperado que não tinha relação positiva com o modelo preditivo. Contudo, variáveis como Blue_firsttower, Blue_firstblood impressionam por ter pouca pontuação no segundo modelo. Ao todo, foram removidas um total de 23 variáveis, restando 24 variáveis no segundo modelo.

4.5 Limpeza dos Dados

Com o intuito de melhorar o desempenho dos algoritmos e evitar conclusões falsas o processo de limpeza de dados tem objetivo de detectar valores inconsistente, valores nulos e valores duplicados, além disso, deve-se analisar e remover possíveis *Outliers* do conjunto de dados.

Também é possível aplicar filtros nas análises de dados. Com a aplicação desses filtros é possível remover ruídos e dados inconsistente no conjunto de dados. Por ser tratar de uma base de dados retirada de partidas de campeonatos oficiais não serão aplicadas condições nas nossas análises.

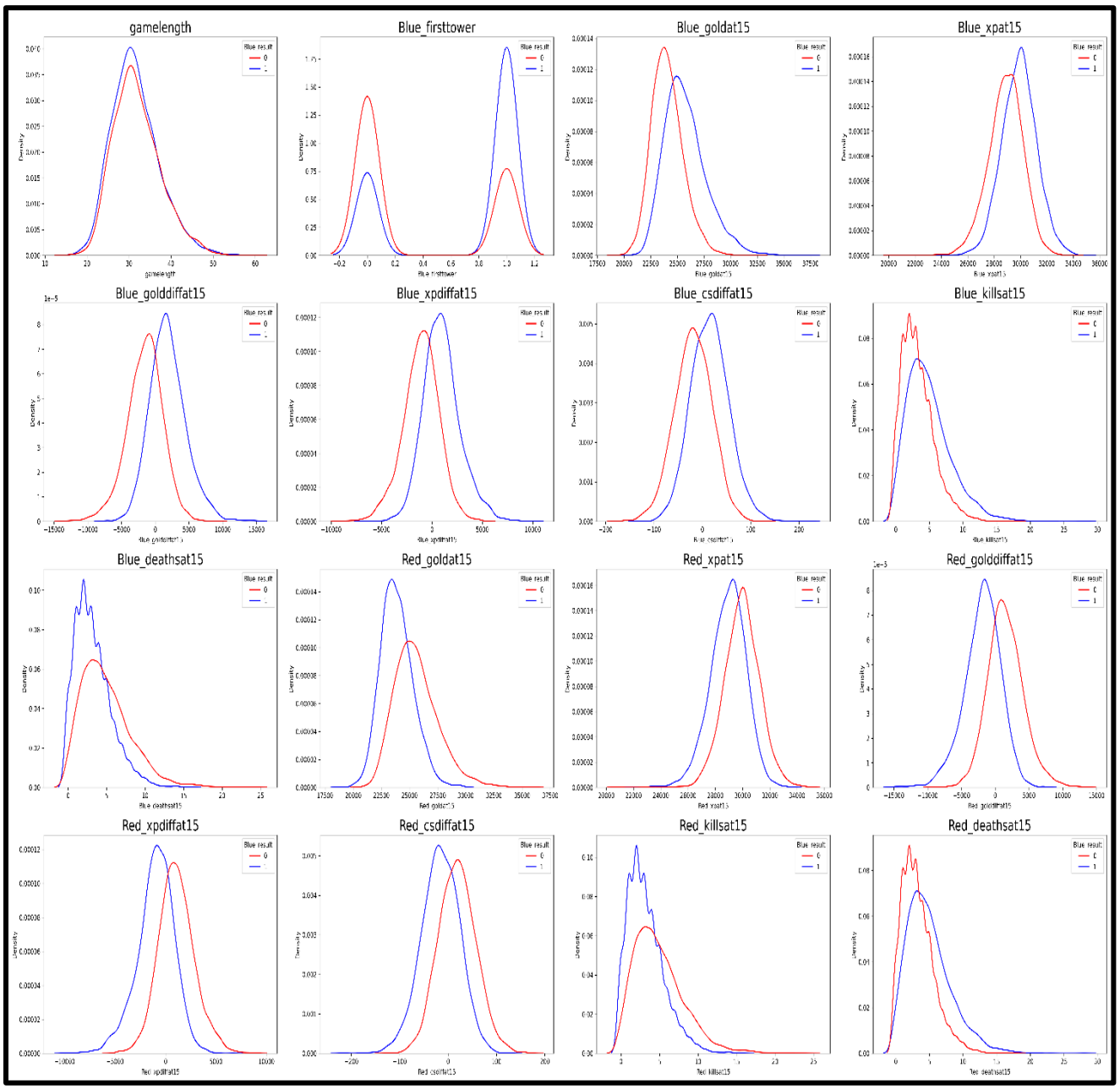
Desta forma, pode-se extrair todos os dados possíveis, todavia, trabalhos futuros que relacione a sua base de dados diretamente às partidas casuais, deve utilizar filtros como no trabalho de Nascimento (2017), que propôs remover partidas rendidas e partidas que contém jogadores que abandonaram.

Se tratando de uma base de dados de campeonatos, não existem jogadores que abandonaram partida, pois isso intitula antijogo passível de multa e até banimento do campeonato. Além disso, não houve partidas rendidas no campeonato, mas sim partidas que acabaram muito rápido, sendo que a partida mais rápida do conjunto de dados durou 15 minutos e a mais longa 59 minutos.

Analisando os valores contidos nos dois modelos, foram verificados a inexistência de valores nulos através do *.isnull().sum(),* além disso, foram verificados o tipo das variáveis *.info()* , quantidade de valores únicos *.nunique()* e valores duplicados .drop_duplicates(), ambas funções da biblioteca pandas. Foram encontrados 10 valores duplicados em ambos os modelos que passaram de 10.585 para 10.575 registros.

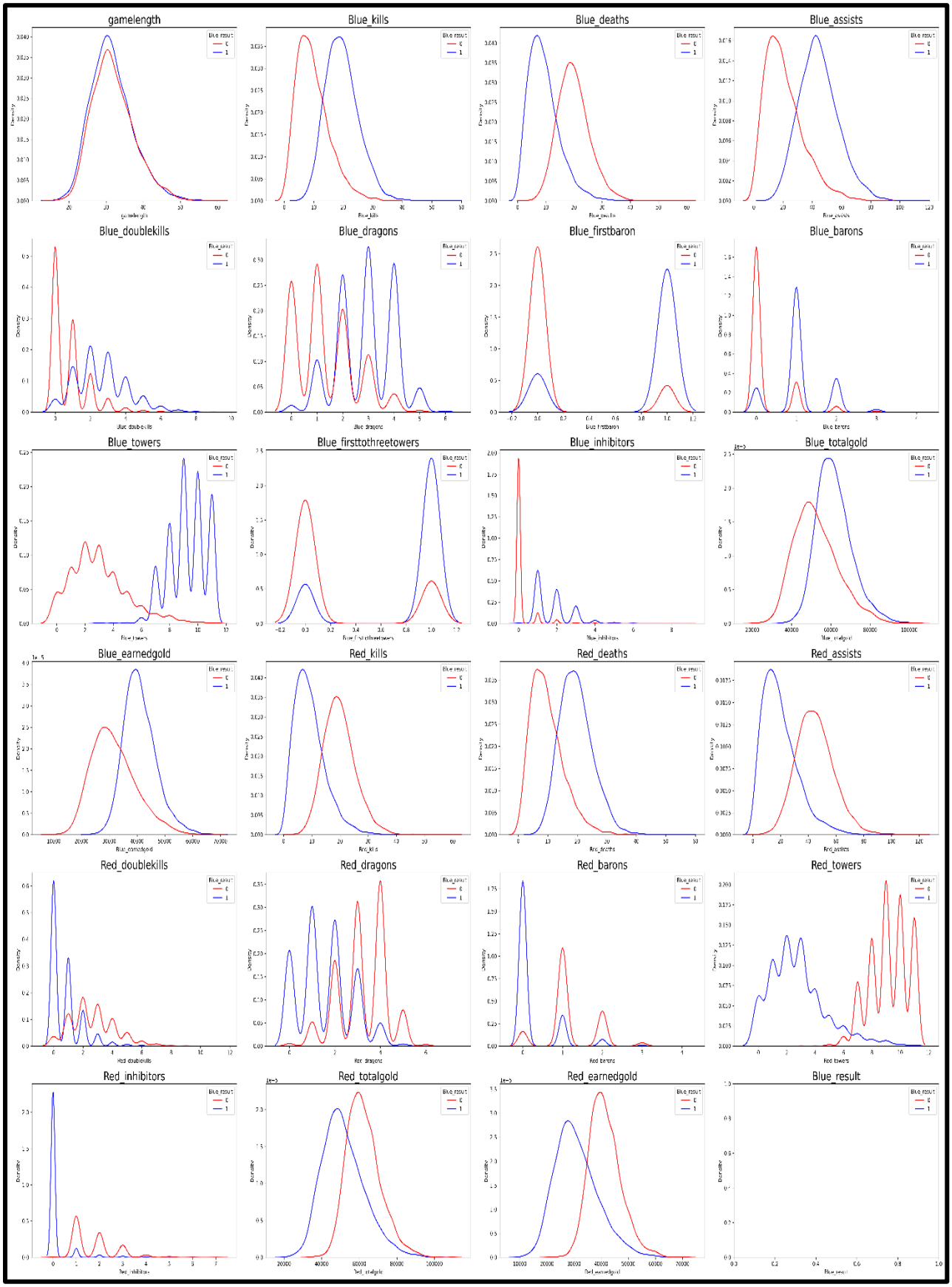
Um importante meio de análise exploratória dos dados é visualizar como os dados estão distribuídos ao longo de um intervalo. As Figuras 17 e 18 mostram os gráficos de distribuição dos dados para o modelo 1 e modelo 2 respectivamente, relacionando o gráfico de distribuição com a variável *target Blue_result:*

Figura 17 - Distribuição dos Dados do primeiro modelo



Fonte: autoria própria.

Figura 18 – Distribuição dos Dados do segundo modelo.



Fonte: autoria própria.

A partir desses gráficos de distribuição é possível analisar o comportamento dos dados através das classes 1 (time azul) e a classe 0 (time vermelho) da variável *target*. Através da análise desse comportamento é possível avaliar se as classes estão balanceadas ou se ocorre certa discrepância entre os registros.

4.5.1 *Outliers*

Os *outliers* são denominados valores extremos que diferencie, significativamente, o padrão dos dados. Esses valores são considerados fora da curva, indicando possível anomalia, erro ou variação natural dos dados. Pode-se ter erros de medição por parte do equipamento, resultados que diferem significativamente do esperado. Além disso, também é possível variações naturais nos dados. Sendo assim, parte do analista de dados verificar essas ocorrências.

A fim de encontrar um modelo preditivo para os modelos, foi adotado a utilização dos métodos de classificação de Regressão Logística, *Random Florest* e Árvore de Decisão para analisar o desempenho de uma partida através dos dados coletados e informar se aquela equipe sairá vitoriosa ou perdedora.

Para realização de um bom modelo preditivo, foi feita uma análise de *outliers* para cada modelo, desta forma, se avalia os possíveis valores que possam afetar significativamente o modelo preditivo. Primeiramente foi feita uma análise descritiva do conjunto de dados, com isso é possível analisar valores como *count*, média, 25%, mediana, 75%, mínimo, máximo e desvio padrão.

Em seguida, observando os gráficos de histograma e boxplots dos modelos, foi aplicado a remoção de valores extremos e posteriormente plotado os gráficos de histograma e boxplots sem os possíveis *outliers.*

4.5.1.1 Análise de *Outliers* para o modelo 1

A estatística descritiva é um importante meio de visualizar valores como count mínimo, média, 25%, mediana, 75%, desvio padrão e valores máximo. Desta forma, é possível saber se existem valores faltando nas variáveis, quais são os valores extremos, entre outros.

A Tabela 4 mostra a estatística descritiva do modelo 1:

Tabela 4 – Análise descritivas das variáveis do primeiro modelo.

	count	***mean***	***desvio padrão***	***min***	***25%***	***50%***	***75%***	***max***
gamelength	10575	31.703237	5.667202	15.35	27.66	31.06	35.08	59.61
Blue_firsttower	10575	0.542600	0.498205	0	0	1	1	1
Blue_goldat15	10575	24947.220709	1951.233330	19241	23588	24693	26038	37279
Blue_xpat15	10575	29406.750544	1426.269679	20358	28502.5	29446	30325.5	35006
Blue_golddiffat15	10575	247.647376	3193.965400	-13460	-1738	211	2190	15041
Blue_xpdiffat15	10575	-6.240662	2094.406262	-9038	-1312	-41	1254	10019
Blue_csdiffat15	10575	-0.689078	42.884727	-175	-29	-1	28	221
Blue_killsat15	10575	4.133995	3.003483	0	2	4	6	28
Blue_deathsat15	10575	4.018629	2.945494	0	2	3	6	24
Red_goldat15	10575	24699.573333	1911.057744	18800	23352	24463	25770	35789
Red_xpat15	10575	29412.991206	1417.431228	20794	28532	29462	30320	34926
Red_golddiffat15	10575	-247.647376	3193.965400	-15041	-2190	-211	1738	13460
Red_xpdiffat15	10575	6.240662	2094.406262	-10019	-1254	41	1312	9038
Red_csdiffat15	10575	0.689078	42.884727	-221	-28	1	29	175
Red_killsat15	10575	4.008700	2.940061	0	2	3	6	24
Red_deathsat15	10575	4.144019	3.008903	0	2	4	6	28
Blue_result	10575	0.522837	0.499502	0	0	1	1	1

Fonte: autoria própria.

Com essa análise descritiva é possível observar os parâmetros dos dados. Além disso, é um importante meio para verificar valores após remoção de dados extremos.

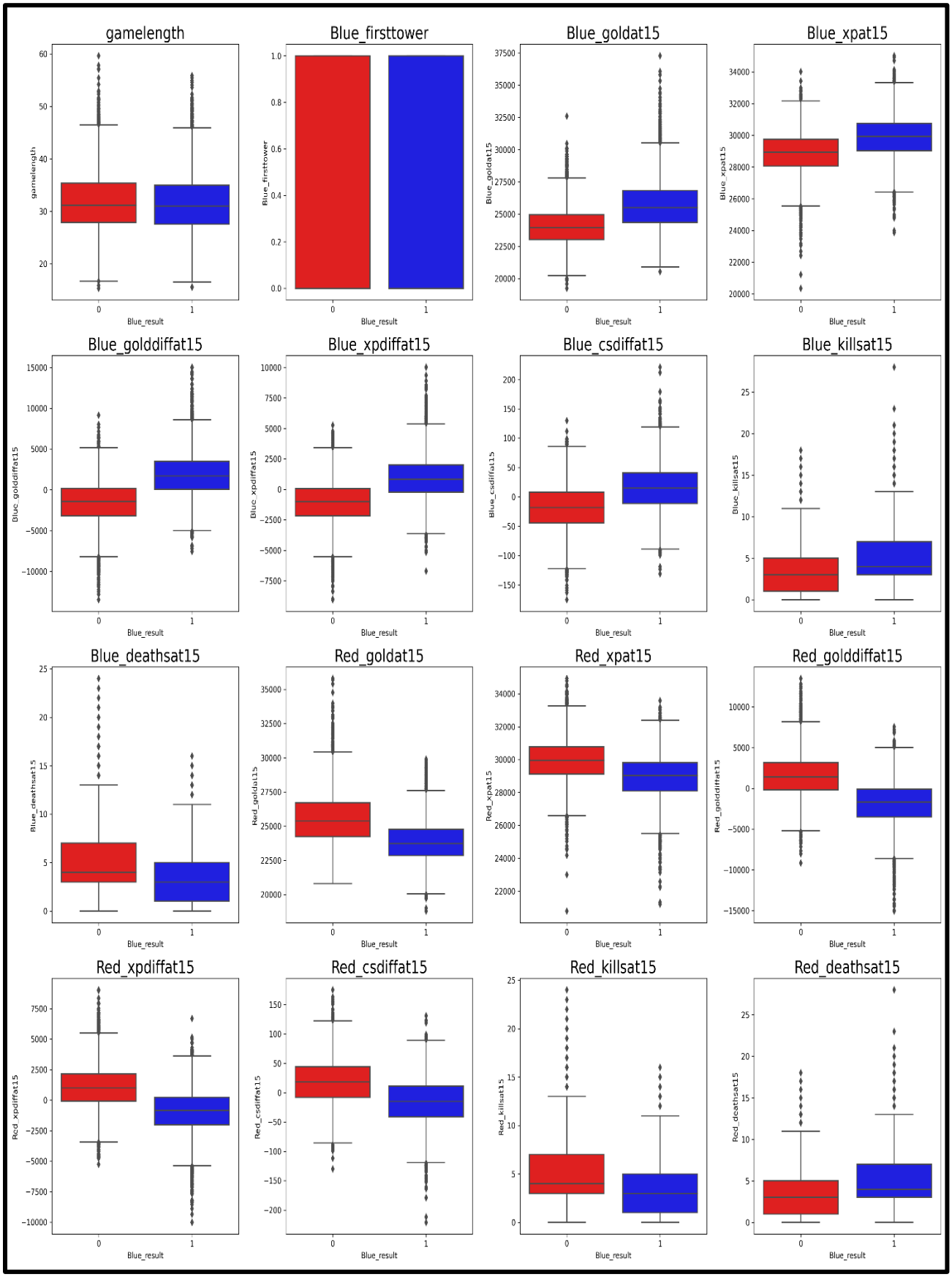
A análise de *outliers* ocorreu através do diagrama de *boxplots* conhecido como ‘gráfico de caixa’ ou ‘diagrama de caixa’. Nela são representados os gráficos em cinco estatísticas resumidas: valor mínimo, primeiro quartil (25%), mediana, terceiro quartil (75%) e valor máximo.

Segundo Montgomery (2014) este gráfico exibe, simultaneamente, os dados de dispersão, tendência central, afastamento da simetria e identificação de observações de valores afastados (valores discrepantes ou *outliers*).

Para verificação do comportamento dos dados: forma, simetria e assimetria dos dados, utilizou-se o gráfico de distribuição de valores chamado histograma. Segundo Nuzzo (2019), histograma são utilizados para transmitir graficamente conhecimento de uma distribuição de dados. Sua estrutura consiste em um gráfico com número pré-determinado de colunas chamadas de *bins*. Por fim, vale ressaltar que número de *bins* e sua largura impactam na visualização final dos dados (NUZZO, 2019).

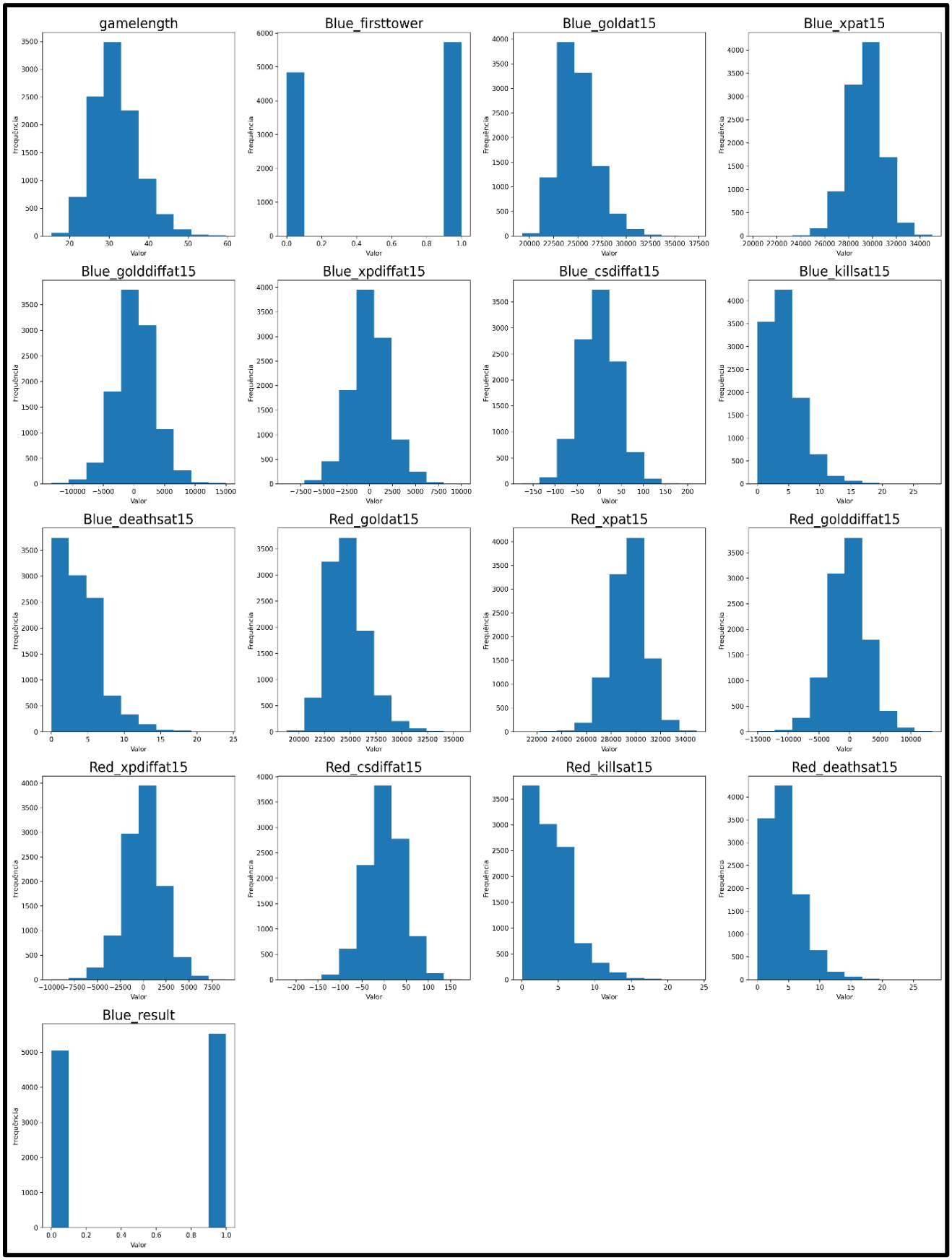
As Figuras 19 e 20 representam, respectivamente, os gráficos de boxplots e de histograma para análise aos 15 minutos de partida:

Figura 19 – *BoxPlot* do primeiro modelo.



Fonte: autoria própria.

Figura 20 – Histograma do primeiro modelo.



Fonte: autoria própria.

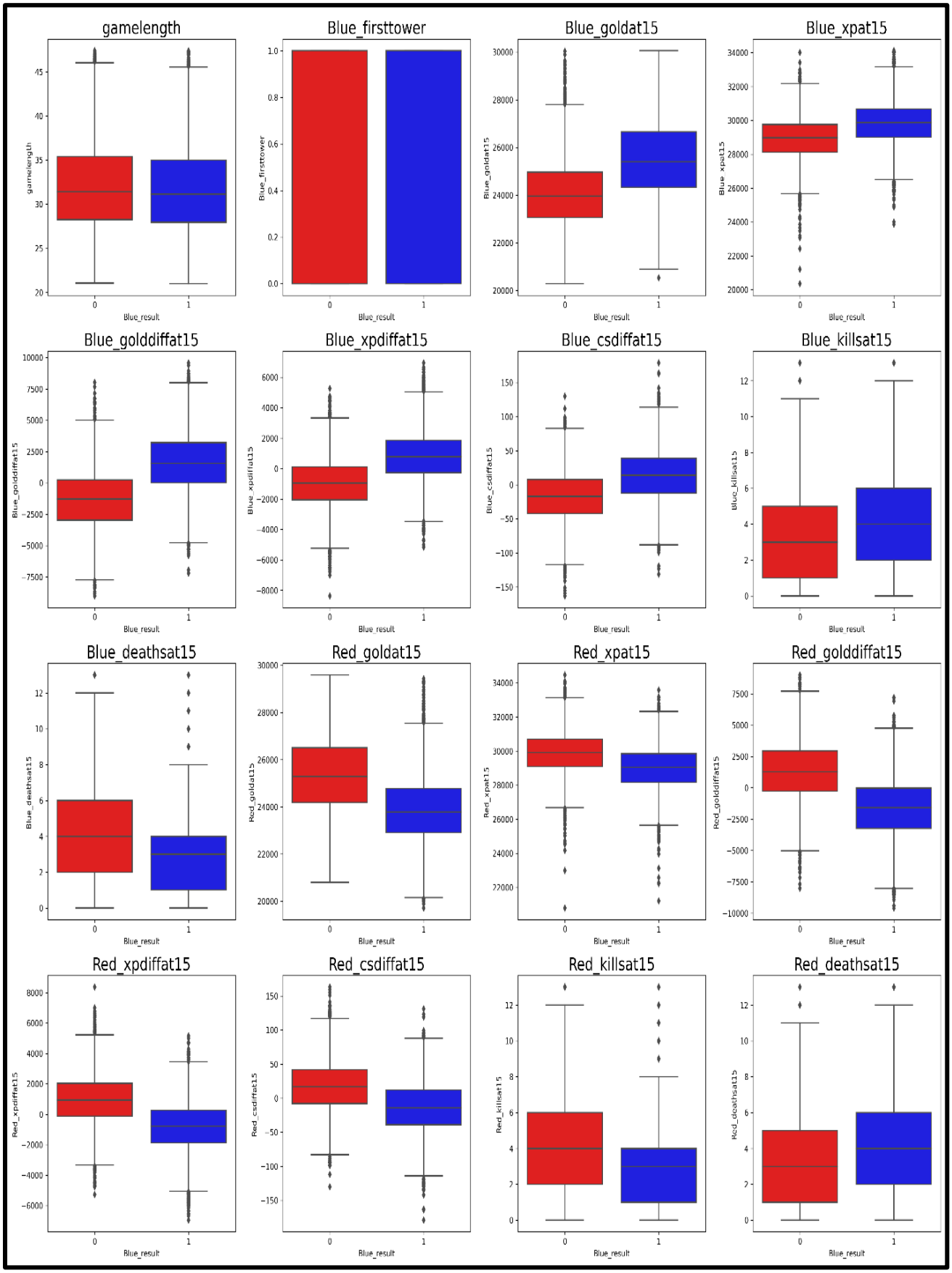
Observando as Figuras 19 e 20, nota-se que existem número de *outliers* dentro do conjunto de dados do *League of Legends.* Dentro do contexto do jogo observa-se a existência de dados que variam em grande escala. Desse modo, como não foi identificada a causa desse comportamento, foi considerado remover apenas *outliers* extremos.

Para remoção *outliers* foi selecionado as variáveis *gamelength, Blue_killsat15, Red_killsat15, Blue_goldat5 e Red_goldat15*, no qual, foram removidos 1% dos seus valores extremos. Através da duração de partida (*gamelength)* é possível remover valores mínimos e máximos de partida que foram rápidas demais, consideradas *stomp* e partidas que demoraram demais, consideradas partidas de *late game.*

Partidas que acabam rápido demais ou que demoram demais acabam influenciando as demais variáveis na hora de criar um modelo preditivo. Da mesma forma, as variáveis *Blue_kilsat15 e Red_killsat15* influenciam. Por exemplo, com partidas que tiverem altas taxas de kills o que aumenta a quantidade de ouro, dano e demais variáveis. Por fim, seguindo a mesma lógica as variáveis *Blue_goldat15 e Red_golat15* influenciam no crescimento das demais variáveis.

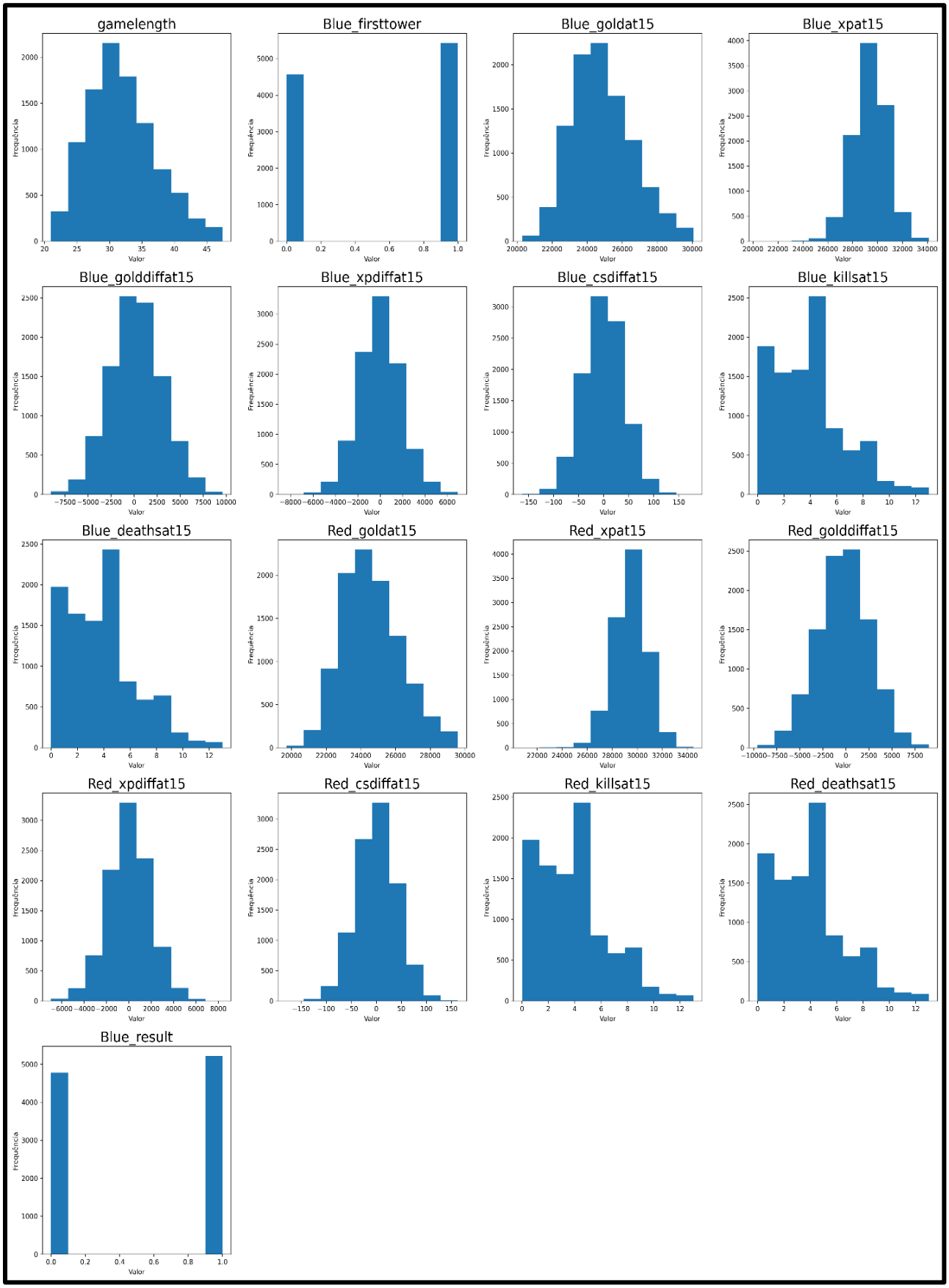
Após remoção desses *outliers* extremos as Figuras 21 e 22 representam graficamente os gráficos de boxplots e o histograma após realização dos valores extremos:

Figura 21 – *BoxPlot* do primeiro modelo após remoção dos *Outliers*



Fonte: autoria própria.

Figura 22 – Histograma do primeiro modelo após a remoção dos *Outliers*



Fonte: autoria própria.

Logo após a remoção dos *outliers*, os valores de mínimo e máximo de duração de partida, saindo de 15 minutos para 21 minutos para os valores de mínimo e de 59 minutos para 47 minutos para valores máximos. Da mesma forma, as demais variáveis tiveram redução conforme as Figuras 21 e 22. Além disso, para este modelo foi removido um total de 582 registros, saindo de 10.575 para 9.993 após remoção dos *outliers*.

4.5.1.2 Análise de *Outliers* para o modelo 2

Assim como feito no modelo 1 a Tabela 5 representa as estatísticas descritivas das variáveis do modelo 2:

Tabela 5 – Análise descritivas das variáveis do segundo modelo

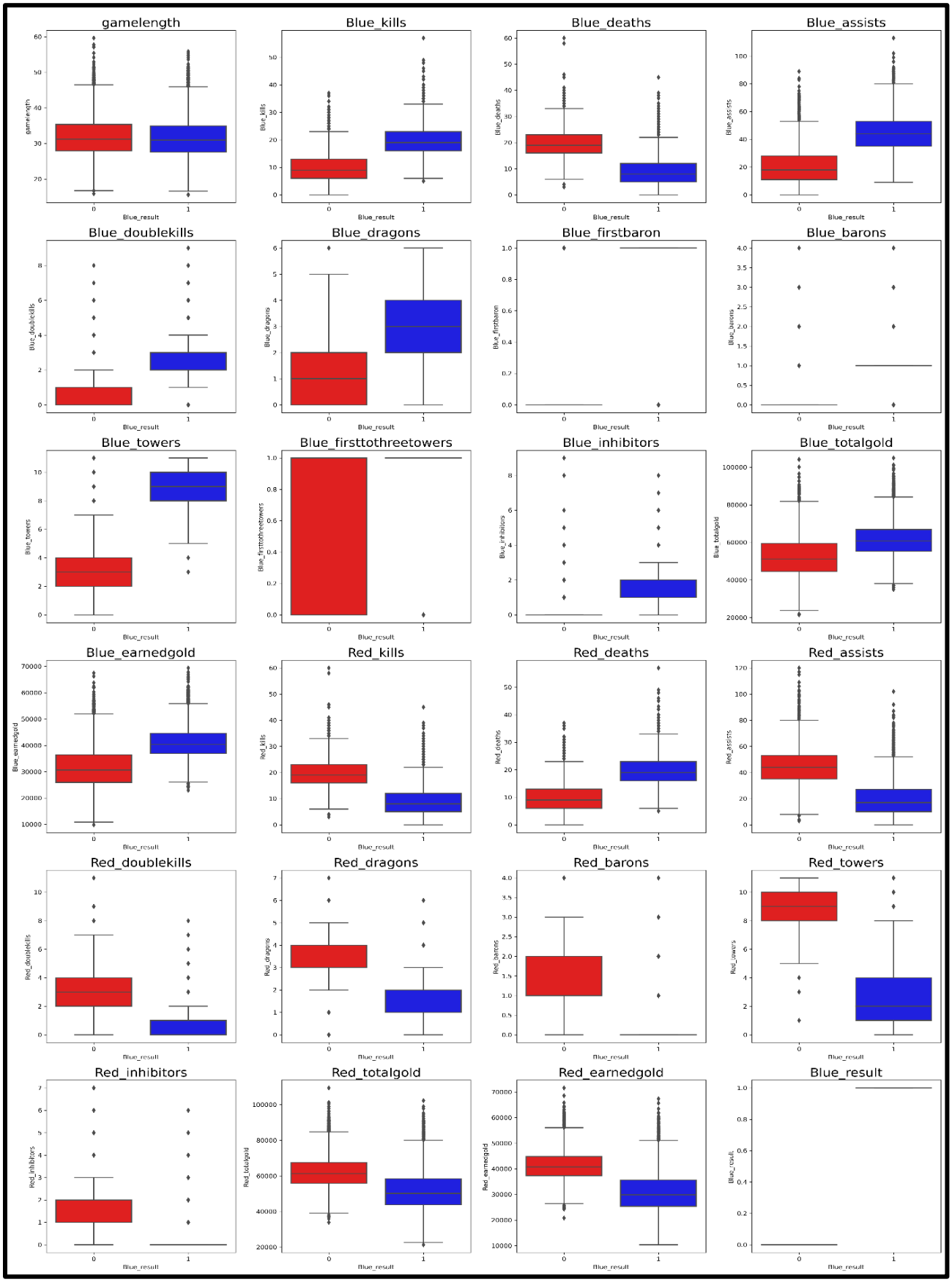
	count	*mean*	*desvio padrão*	*min*	*25%*	*50%*	*75%*	*max*
gamelength	10575	31.70	5.66	15.55	27.68	31.06	35.08	59.61
Blue_kills	10575	14.871678	7.474886	0	9	15	20	57
Blue_deaths	10575	14.357447	7.677555	0	8	14	20	60
Blue_assists	10575	33.208700	17.807910	0	19	33	46	113
Blue_doublekills	10575	1.727470	1.578792	0	0	1	3	9
Blue_dragons	10575	2.141087	1.371006	0	1	2	3	6
Blue_firstbaron	10575	0.478487	0.499561	0	0	0	1	1
Blue_barons	10575	0.669787	0.707501	0	0	1	1	4
Blue_towers	10575	6.274232	3.566238	0	3	7	9	11
Blue_firsttothreetowers	10575	0.544397	0.498049	0	0	1	1	1
Blue_inhibitors	10575	0.979669	1.101464	0	0	1	2	9
Blue_totalgold	10575	57350.568227	11169.278219	21496	49825	57047	64371	104925
Blue_earnedgold	10575	36568.612861	8412.055400	9867	30671	37072	42127.5	69387
Red_kills	10575	14.326998	7.678481	0	8	14	20	60
Red_deaths	10575	14.900709	7.473525	0	9	15	20	57
Red_assists	10575	31.692009	18.261503	0	16	31	45	120
Red_doublekills	10575	1.659574	1.613099	0	0	1	3	11
Red_dragons	10575	2.356028	1.383061	0	1	2	3	7
Red_barons	10575	0.682175	0.741817	0	0	1	1	4
Red_towers	10575	5.834515	3.652827	0	2	6	9	11
Red_inhibitors	10575	0.890591	1.089108	0	0	1	1	7
Red_totalgold	10575	56738.653428	11565.551471	21344	48827.5	56508	64139.5	109439
Red_earnedgold	10575	35956.698061	8776.863677	10398	29330	36531	41952	71576
Blue_result	10575	0.522837	0.499502	0	0	1	1	1

Fonte: autoria própria.

Com base nessa análise descritiva é possível observar valores como por exemplo a média de duração de partida, representado pelo valor de 31 partidas, além disso, um número muito alto de *kills*, sendo 57 para a variável *Blue_kills* e 60 para a variável *Red_kills*. Outro fator que impressiona é jogos com valores máximos de 113 assistências para a variável *Blue_assists* e 120 assistências para a variável Red_assists.

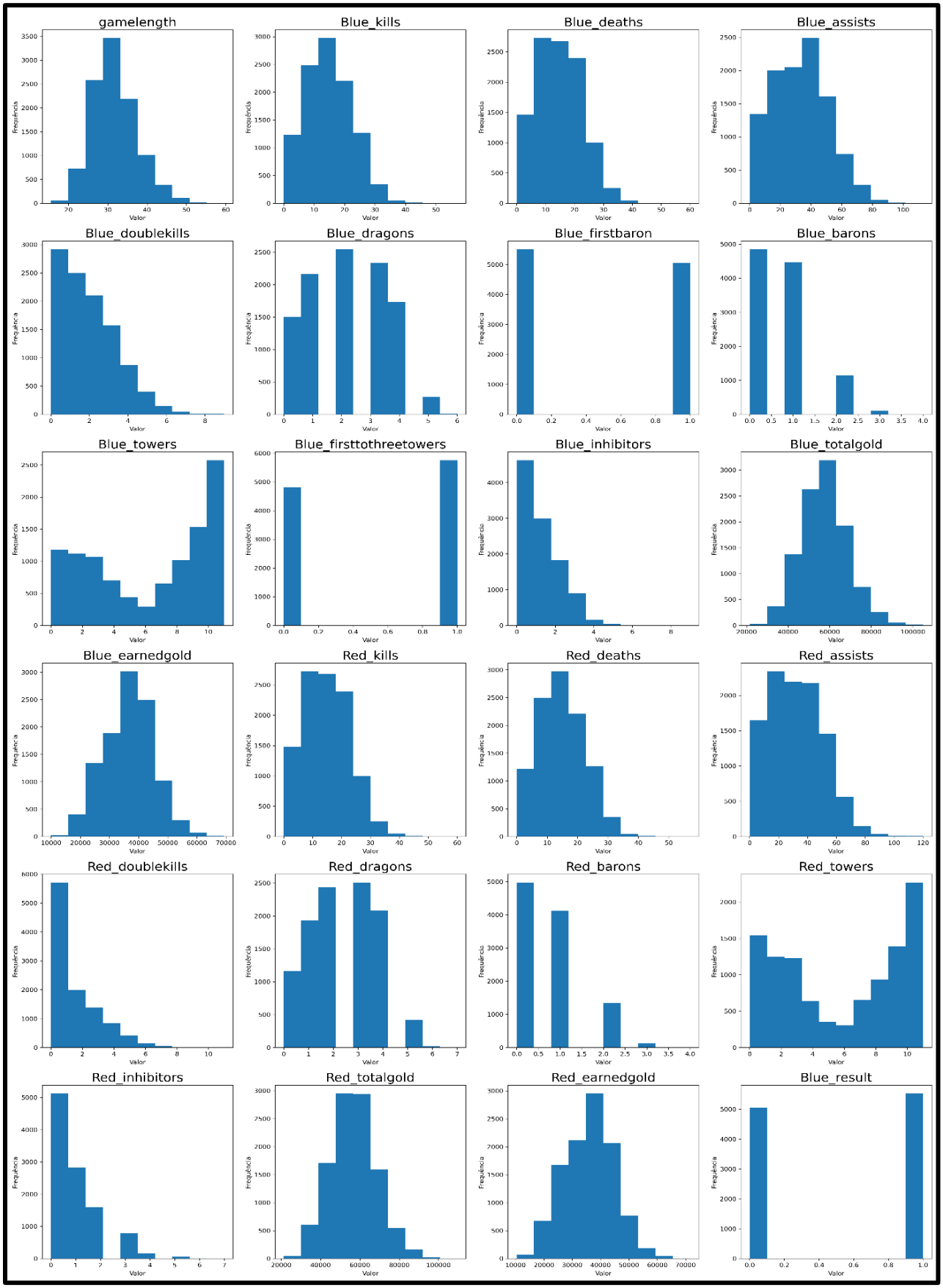
Uma forma de visualizar esses possíveis outliers é através dos gráficos de boxplots e histograma, desta forma, as Figura 23 e 24 representam esses gráficos respectivamente para análise dos dados ao final de jogo:

Figura 23 – *BoxPlot* do segundo modelo



Fonte: autoria própria.

Figura 24 – Histograma do segundo Modelo

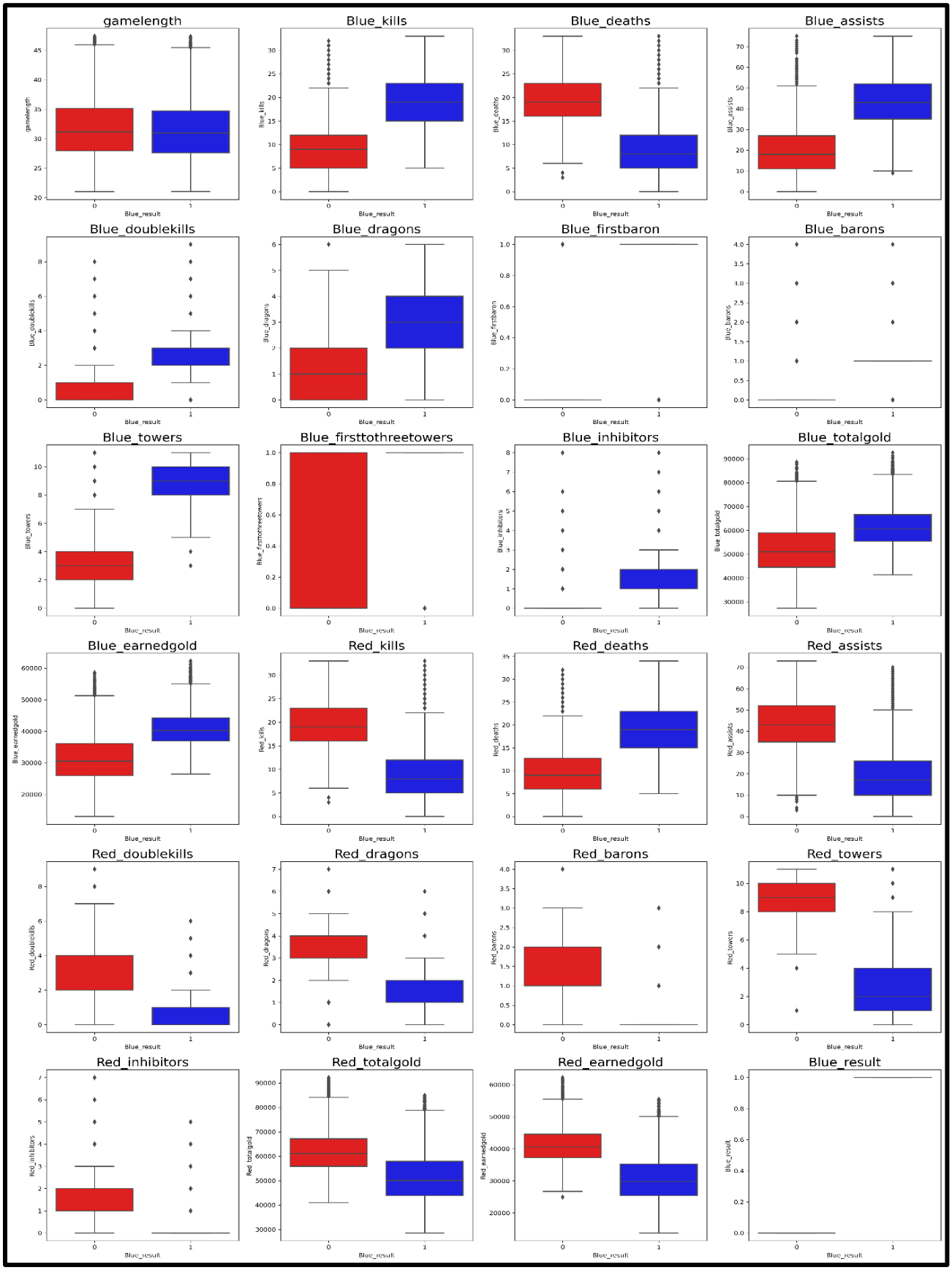


Fonte: autoria própria.

Assim como foi feito no primeiro modelo, não sabemos ao certo o motivo de tanto outliers. Desta forma, foi removido valores extremos de outliers, selecionando as variáveis *gamelength, Blue_kills, Red_kills, Blue_assists e Red_assists*, no qual, foram removidos 1% dos seus valores extremos.

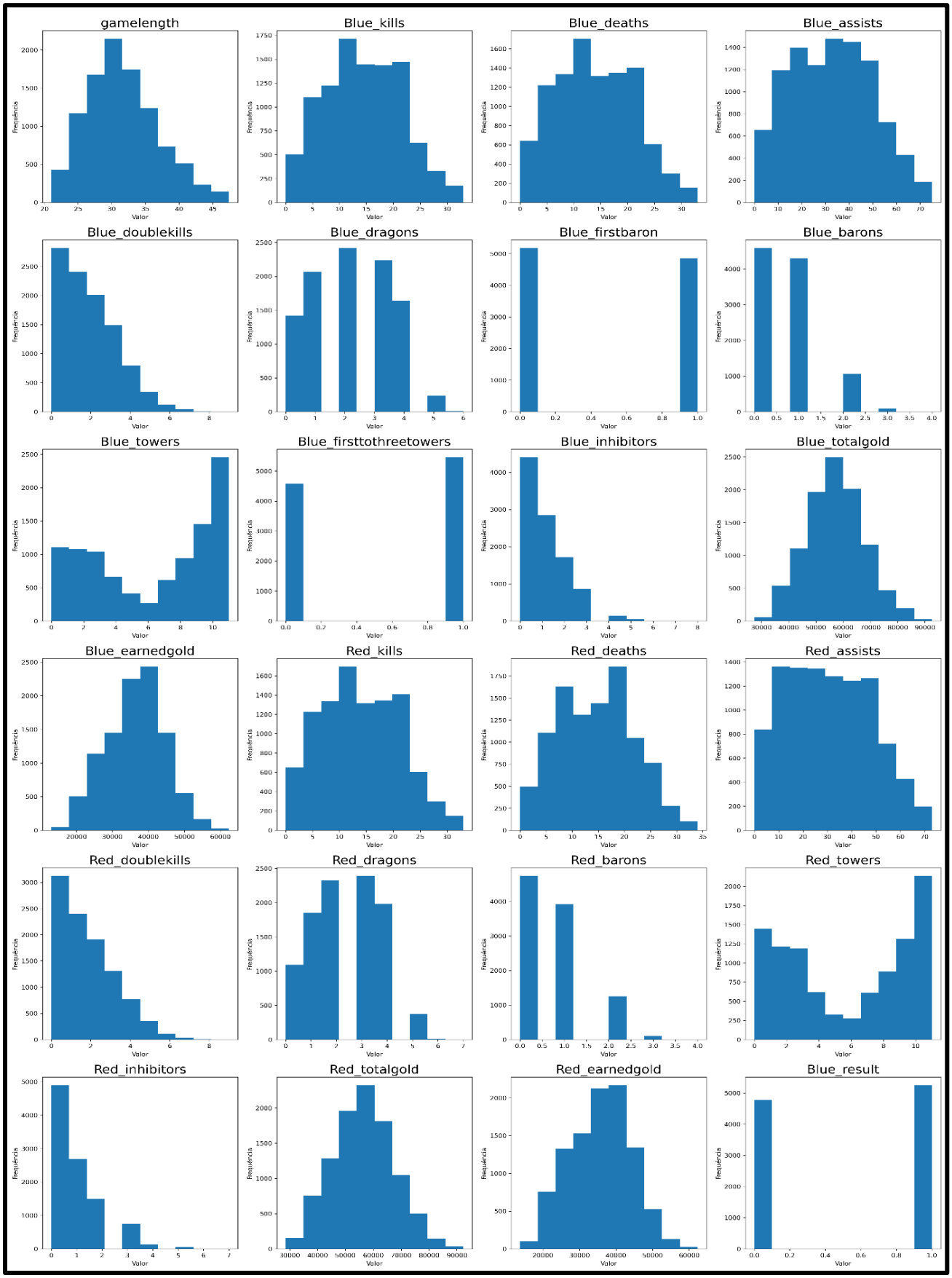
Vale ressaltar, que da mesma forma que ocorreu no primeiro modelo, a duração da partida influência o modelo. Desta forma, analisar esta variável em relação às variáveis de finais de jogo é de extrema importância, pois pode ocorrer discrepância dos dados. Após a remoção desses *outliers* extremos as Figuras 25 e 26 demonstra os gráficos boxplots e o histograma respectivamente após realização desta etapa:

Figura 25 – *BoxPlot* do segundo modelo após remoção dos *Outliers*



Fonte: autoria própria.

Figura 26 – Histograma do segundo modelo após remoção dos *Outliers*



Fonte: autoria própria.

Logo após essa etapa, os valores iniciais de partida sendo igual a 21 minutos e valores máximos de 47 minutos conforme o primeiro modelo. Valores máximos das variáveis *Blue_kills e Red_kills* foram reduzidos de para no máximo 33 kills para as duas variáveis. Além disso, as variáveis *Blue_assists* foram reduzidas para um máximo de 75 assistência e a variável *Red_assists* para 73 assistências.

É importante salientar, que mesmo removendo valores extremos somente para essas variáveis, as demais variáveis também tiveram redução, conforme mostrado nas Figuras 25 e 26. Em síntese, nesta etapa o segundo modelo foi removido um total de 544 registros, saindo de 10.575 para 10.031 após remoção dos *outliers*.

4.6 Transformação dos dados

Essa seção descreve os processos de transformação dos dados, realizando cálculo de métricas para o segundo modelo e a normalização dos dados pelo método min-max para ambos os modelos.

4.6.1 Cálculo de métricas

Em jogos como o *League of Legends*, o tempo pode variar entre as partidas não tendo um tempo mínimo acabar e nem um tempo máximo. Desta forma, o cálculo de métricas propõem normalizar os dados pela duração da partida, sendo possível analisar os desempenhos de várias equipes independentemente da duração. Este também é um processo que antecede a análise de *SelectKBest e* correlações entre as variáveis.

O conjunto de dados X, deve-se dividir cada desempenho de cada equipe x pela duração da partida que a equipe participou:

$$\frac{X}{mDuração} \tag{16}$$

Após essa etapa, existem métricas de desempenho de equipes. Por exemplo, abates por minutos (*kill*), assistência por minutos (*assists), mortes por minuto, entre outras.* Em síntese a etapa de cálculo de métricas foi feita apenas para segundo modelo, pois o primeiro modelo já tem pré-definido um tempo máximo para analisar os dados em uma mesma métrica.

4.6.2 Normalização dos Dados

A normalização dos Dados é um processo que define um intervalo de valores através de métodos.

Dentro dos dois modelos estudados do conjunto de dados do *League of Legends,* percebe-se intervalo dos valores variando amplamente entre as características. Portanto, foi aplicado a normalização dos dados *min-max*, no qual, aplica-se a proporcionalidade em uma mesma escala [0-1], facilitando comparar valores redundantes em ambos os modelos preditivos. A fórmula da normalização por *min-max* pode ser representada a na Equação 17:

$$x'_{\mathrm{j}} = \frac{x_{\mathrm{ji}} - \min\,(X_{\mathrm{j}})}{\max(X_{\mathrm{j}}) - \min\,(Xj)} \qquad (17)$$

Onde:

- x_{ji} é o valor original que está sendo normalizado.
- $\min\,(X_{\mathrm{j}})$ é o valor mínimo da variável.
- $\max(X_{\mathrm{j}})$ é o valor máximo da variável.
- x'_{j} valor resultante normalizado, após a aplicação da fórmula.
- $i\ e\ j$ índice *i* representa a posição da linha e o índice *j* representa a posição da coluna.

Após a normalização dos dados, as características do conjunto de dados X assumem valores na faixa [0,1] para ambos os modelos.

4.7 Análise de variáveis altamente correlacionadas

De modo a evitar ruídos e aprimorar os modelos preditivos, bem como aprimorar o agrupamento dos dados e uma facilitar a visualização dos dados conduziu-se a utilização da análise de correlação. Essa análise ocorreu através da utilização da função .corr() da biblioteca pandas, utilizando uma análise de correlação de *Pearson.* Entretanto, é possível selecionar os tipos de análises de como: correlações de *Kendall* e *Spearman.*

Com base no teste de correlação de *Pearson,* foi considerado um par de características de alta correlação são redundantes para um modelo preditivo. Desta

forma, essas variáveis contribuem muito pouco para o modelo preditivo e, as vezes, até proporciona grande aumento computacional.

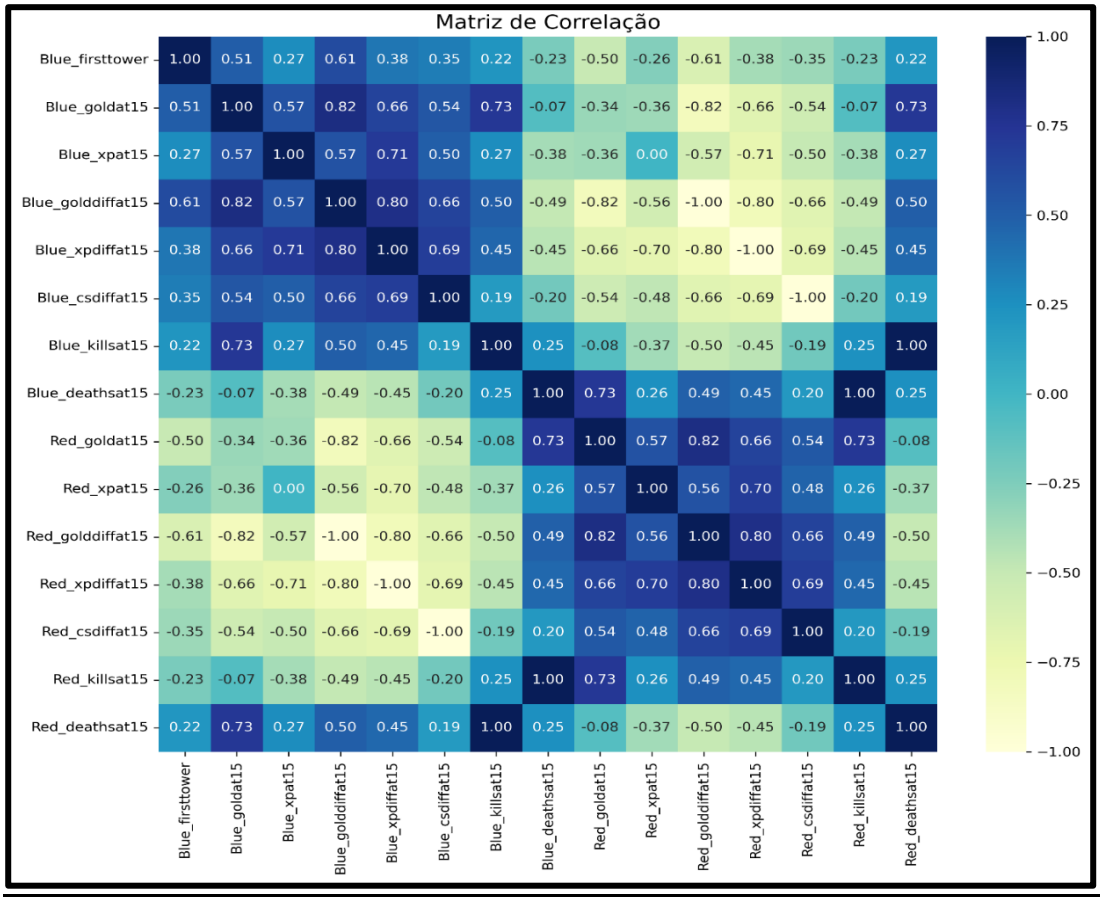
Segundo Damásio (2021), coeficiente de correlação de Pearson ou r de Pearson é uma técnica que faz a métrica se duas variáveis estão relacionadas de maneira linear. Essa relação significa que o aumento de uma variável está relacionado ao aumento da outra, na mesma proporção.

Desse modo, dado um par de características do conjunto de dados altamente correlacionadas $Corr \geq 0.8$, foi selecionada a característica que tem a maior correlação no par, ou seja, a característica em que a soma de todas as suas correlações é maior entre o par de variáveis. Essa seção aborda a análise de correlação para dois modelos, mostrando as variáveis removidas do *dataset* e qual o resultado por trás dessa análise.

4.7.1 Análise de correlação do primeiro modelo

A Figura 27 representa o gráfico de correlação das características de desempenho das equipes no modelo de análise dos 15 minutos de partida. Cada um dos valores representa a correlação de um par de variáveis do conjunto de dados X. A escala de cores indicada ao lado direito do gráfico aponta a intensidades dessas correlações, ou seja, as cores em tom de azul indicam correlações positivas, da mesma forma, as cores no tom de verde claro indicam valores de correlações negativas.

Figura 27 – Matriz de Correlação do primeiro modelo



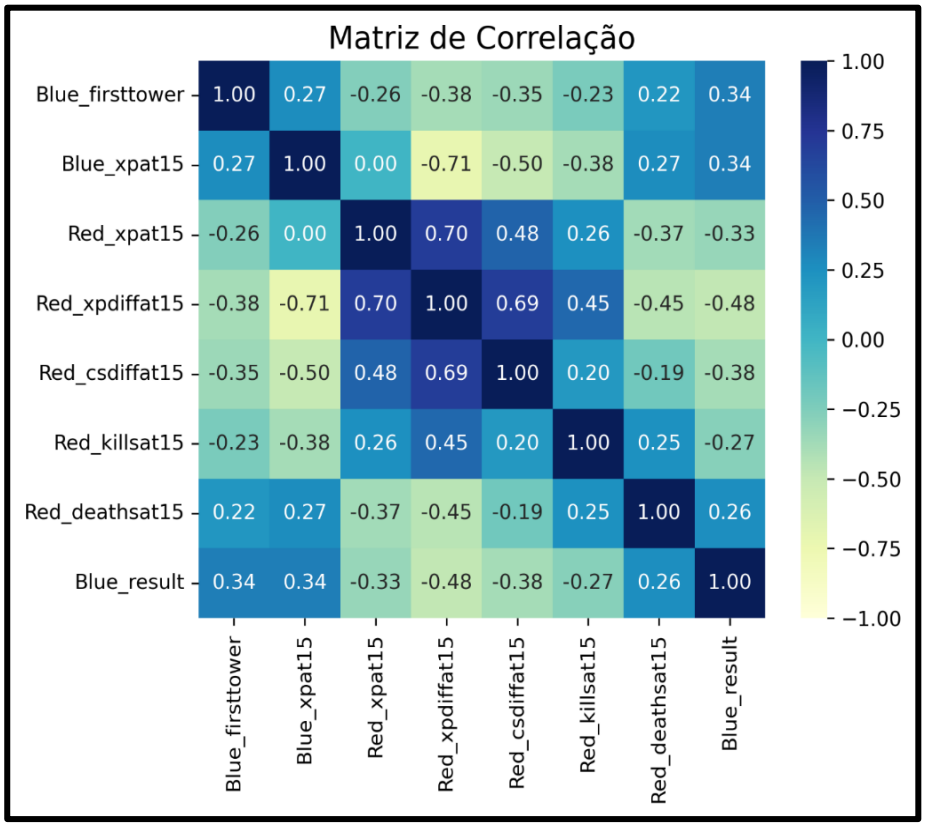
Fonte: autoria própria.

De acordo com a Figura 27 é possível observar correlações, por exemplo, entre as variáveis *Blue_golddiffat15* com a *Red_golddiffat15* sendo uma correlação altamente negativa de – 1.00, além disso, outras variáveis como *Red*. Isso ocorre por conta dessas variáveis serem responsável responsáveis pela diferença. Outras variáveis que possuem correlação negativas são: *Blue_csdiffat15, Red_csdiffat15, Blue_xpdiffat15 e Red_xpdiffat15*.

Neste caso, como essas variáveis possuem o mesmo valor de correlação absoluto sobre as variáveis, foram selecionadas as variáveis de categoria *Blue* e feita a exclusão das variáveis *Red*. Outro fator são as correlações positivas como as variáveis *Blue_killsat15* e Red_deathsat15 que possui uma correlação de 1.00, considerado altamente positiva.

Feito o teste de correlação, as seguintes variáveis altamente correlacionadas foram removidas do primeiro modelo: *Blue_goldat15, Blue_killsat15, Blue_deathsat15, Red_goldat15, Red_golddiffat15, Red_xpdiffat15, Red_csdiffat15.* A Figura 28 mostra a matriz de correlação resultante.

Figura 28 – Matriz de correlação resultante do primeiro modelo



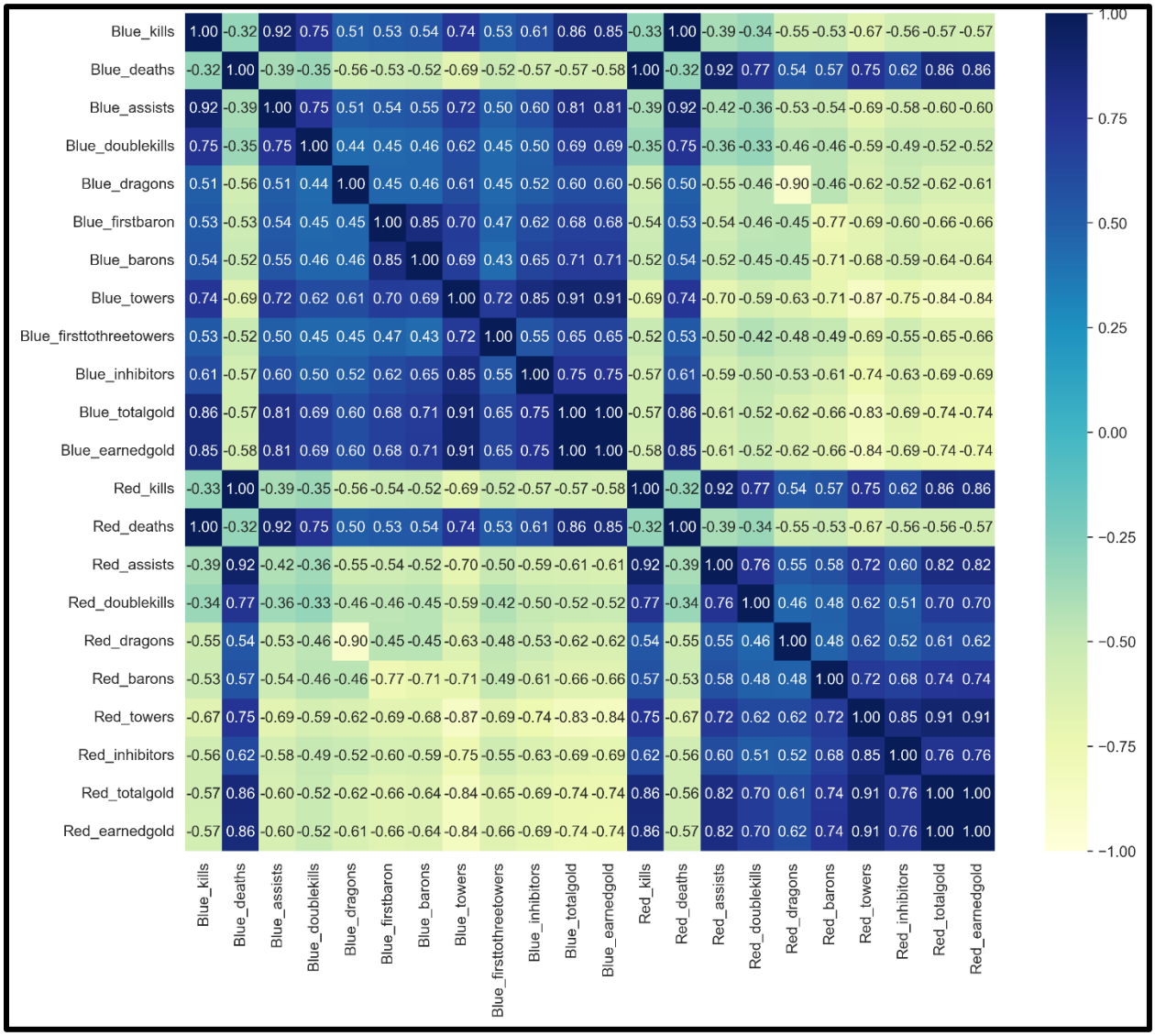
Fonte: autoria própria.

4.7 Análise de correlação do segundo modelo

A Figura 29 representa a matriz de correlação das características de desempenho das equipes no modelo de análise ao final da partida.

Vale ressaltar que, para ambos os modelos, a variável target *Blue_result* incialmente foi removida da primeira análise e colada somente na matriz de correlação resultante.

Figura 29 – Matriz de correlação do segundo modelo



Fonte: autoria própria.

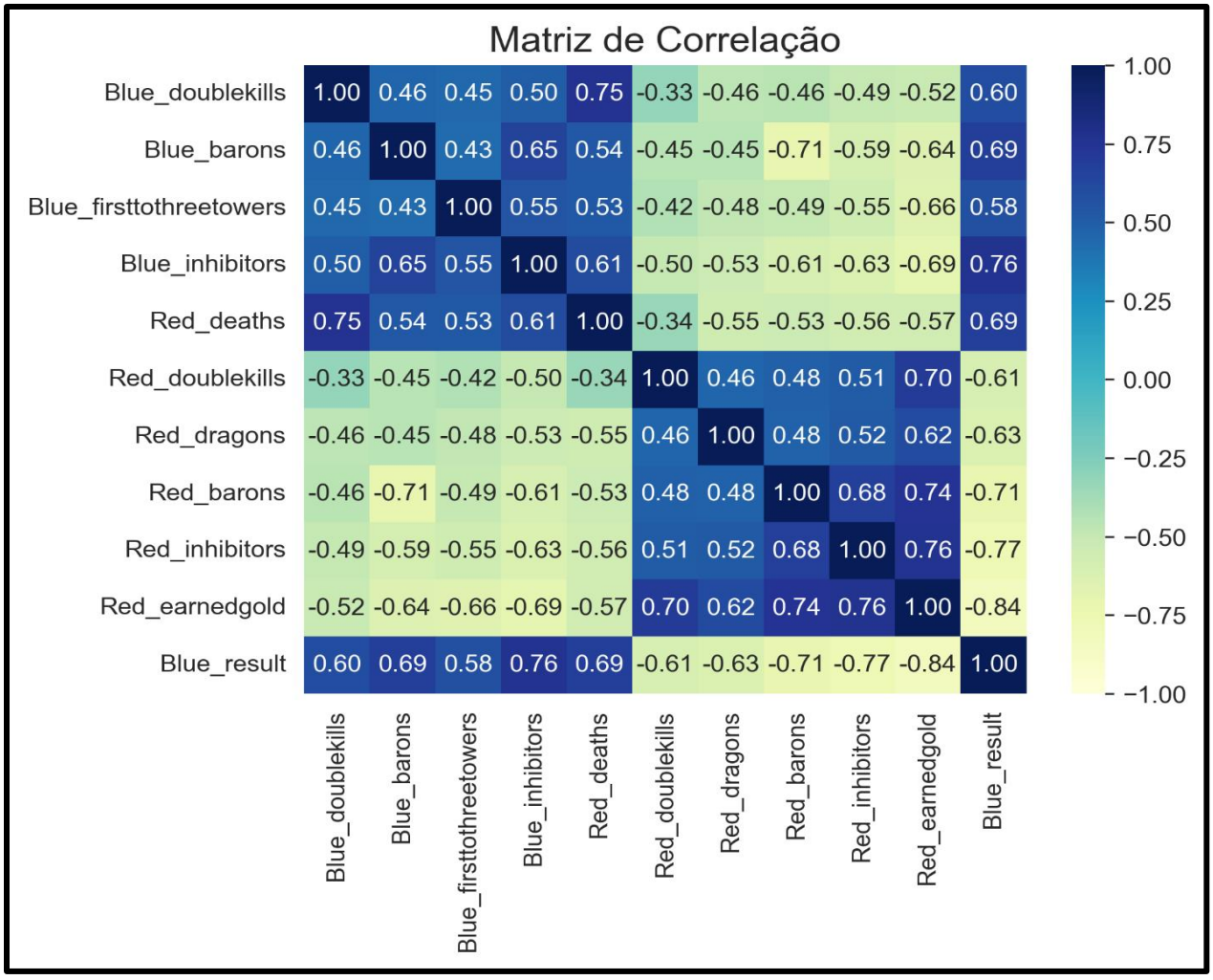
Analisando esse segundo modelo é possível notar correlações entre as variáveis. Por exemplo, correlações entre as variáveis *Blue_totalgold* com a variável *Blue_kills*. Ou seja, quando um jogador consegue um abate em algum momento do jogo, automaticamente recebe uma recompensa em *gold.* Analisando mais um pouco é possível notar diversas correlações entre as variáveis totalgold, de ambas as equipes.

Da mesma forma que a variável Blue_kills está correlacionando positivamente à variável Blue_totalgold, analisando o outro lado da moeda esta mesma variável correlaciona negativamente a variável *Red_kills*. Isso ocorre por conta do *League of Legends* ser um jogo de obtenção de recursos. Desta forma, quando um time obtém um recurso automaticamente ele está negando o recurso do time adversário e vice e versa.

Assim, como no primeiro modelo, variáveis como *Blue_kills* e *Red_deaths* produzem uma correlação igual a 1.00, assim como *Red_kills* e *Blue_deaths*. Desta

forma, deve-se fazer uma análise para identificar a variáveis mais correlacionadas e ser retiradas do modelo. Após realização dos testes de correlação as seguintes variáveis foram removidas do segundo modelo: *Blue_kills, Blue_deaths, Blue_assists, Blue_dragons, Blue_firstbaron, Blue_towers, Blue_totalgold, Blue_earnedgold, Red_kills, Red_assists, Red_barons, Red_towers e Red_totalgold.* A Figura 30 mostra a matriz de correlação resultante.

Figura 30 – Matriz de correlação resultante do segundo modelo



Fonte: autoria própria.

5 ANÁLISE DOS RESULTADOS OBTIDOS E DISCUSSÃO

Este capítulo apresenta os experimentos para os modelos propostos. Com o intuito de analisar e avaliar os resultados obtidos pelos algoritmos propostos para cada modelo, este capítulo está dividindo entre configuração do ambiente experimentos, análise e resultados do primeiro modelo e resultados do segundo modelo.

5.1 Configuração do ambiente de teste

Após realização da etapa de preparação dos dados nos modelos foi possível obter um estudo para aplicabilidade nos modelos de mineração de dados. Desta forma, utilizando uma abordagem de aprendizagem de máquina supervisionada em que os classificadores indicam o vencedor da partida no contexto do *League of Legends*.

As etapas de mineração de dados foram realizadas utilizando o *Python* no ambiente *Jupyter* Notebook e o *software* WEKA. A ferramenta WEKA serviu como base de validação dos resultados. Para isso, o *dataset* foi transformado para um arquivo arff para tornar possível a verificação no software. Além disso, por não apresentar resultados graficamente, foi utilizado o *Jupyter* ao invés do WEKA para representar os resultados dos modelos.

No Jupyter foram utilizadas as bibliotecas de sklearn para importação dos modelos de Regressão logística, Random Florest e Árvore de Decisão. A divisão do *dataset* ocorreu utilizando a *10Fold* Cross-Validation (validação cruzada). A validação cruzada divide o conjunto de dados em 10 partes, no qual, utiliza0se 9 partes para treinamento e uma para teste, mudando a cada rodada os valores de testes e treinamento até completar o clico de 10*Fold.*

Além disso, foram importadas métricas de avaliação do modelo como matriz de confusão, precisão, acuraria, f1_score, entre outros. Por fim, também foram utilizadas bibliotecas, tais como matplotlib e seaborn, para visualização dos resultados obtidos nos testes.

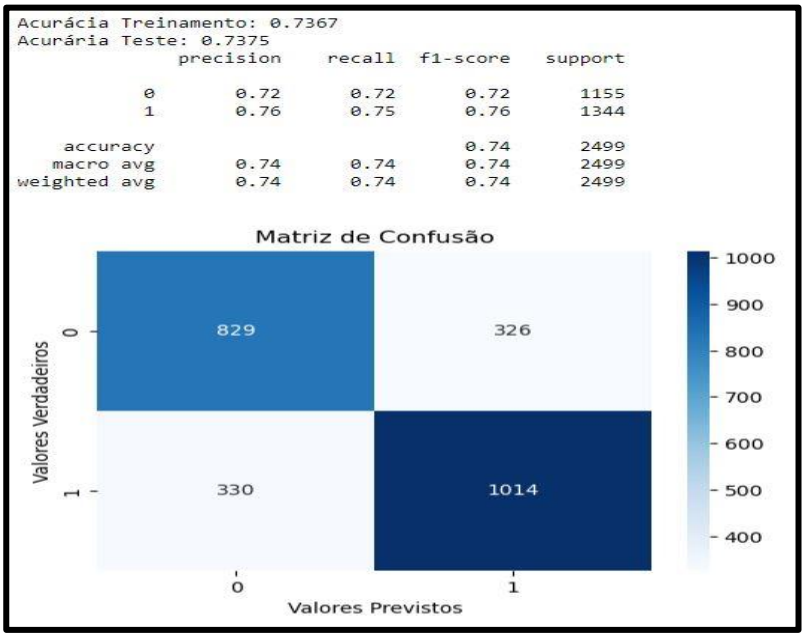
5.3 Resultados e Análises do Primeiro Modelo

O primeiro modelo conta com 9.993 registros e um total de 8 variáveis preditoras. Desta forma, foi utilizado o método 10 Fold Cross-Validation (Validação Cruzada) no

conjunto dos dados. Este método tem uma maior capacidade de generalização dos modelos de aprendizado de máquina.

O primeiro algoritmo analisado foi o de Regressão Logística. A Figura 31 mostra os resultados obtidos:

Figura 31 – Resultados obtidos da Regressão Logística para o primeiro modelo



Fonte: autoria própria.

A regressão logística classificou corretamente em sua fase de treinamento um total de 73,67%. Contudo, na fase de teste, houve uma pequena melhora para 73,75% de classes classificadas corretamente. Com a utilização da função do *classification_report* do *scikit-learn* é possível analisar várias métricas de avaliação para a fase de teste, por exemplo*, precision, recall, f1-score e support.*

Através do *classification_report* é possível analisar cada uma dessas métricas para cada classe, além da média descrita pelo *macro avg* e a média ponderada pela weighted avg. A Figura 31 mostra os melhores resultados de classificação para a classe 1 em relação a classe 0. Ou seja, o modelo de regressão logística classifica melhor para esse conjunto de dados partidas, no qual o time azul ganha, sendo notório através de, por exemplo, a precisão da classe 1 equivalente a 76% e a precisão para a classe 2 igual a 72% da precisão.

Outras métricas analisadas da Figura 31 refere-se ao *recall, F1-score e support.* O *recall* é a taxa de verdadeiros positivos que nosso modelo conseguiu obter. Avaliando

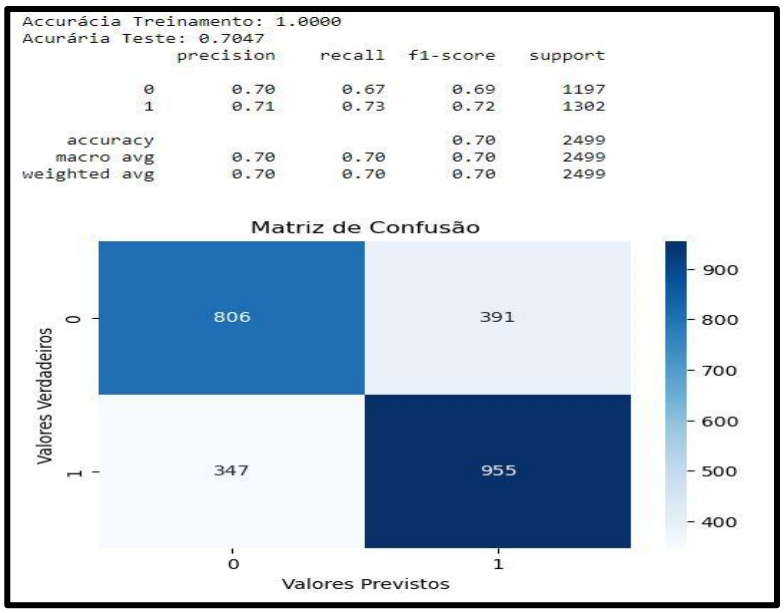
o modelo constatou-se que a média para ambas as classes foi de 74%. *F1-score* é a média harmônica entre *recall e precision*. Desta forma, o modelo conseguiu uma média de 74% e, por fim, o *support* representa os números de ocorrência de cada classe. Sendo assim, fica evidente que existe maior ocorrência para a classe 1.

A matriz de confusão é uma forma visual de verificar a quantidade de vezes que o algoritmo classificou corretamente as classes representado pela sua diagonal principal. A Figura 31 mostra um total de 829 partidas classificadas corretamente para o time vermelho e 326 partidas que o vencedor era o time azul e ele classificou como o time vermelho, sendo um falso positivo.

Da mesma forma, o modelo classificou um total de 1014 partidas corretamente quando o time azul ganhava e 330 partidas ele classificou como sendo o time azul que ganhava, quando na verdade era o time vermelho. Através da matriz de confusão fica evidente que houve melhores acertos para o time azul.

O segundo algoritmo analisado foi o de Random Florest. A Figura 32 mostra os resultados obtidos para este modelo:

Figura 32 – Resultados obtidos do *Random Florest* para o primeiro modelo



Fonte: autoria própria.

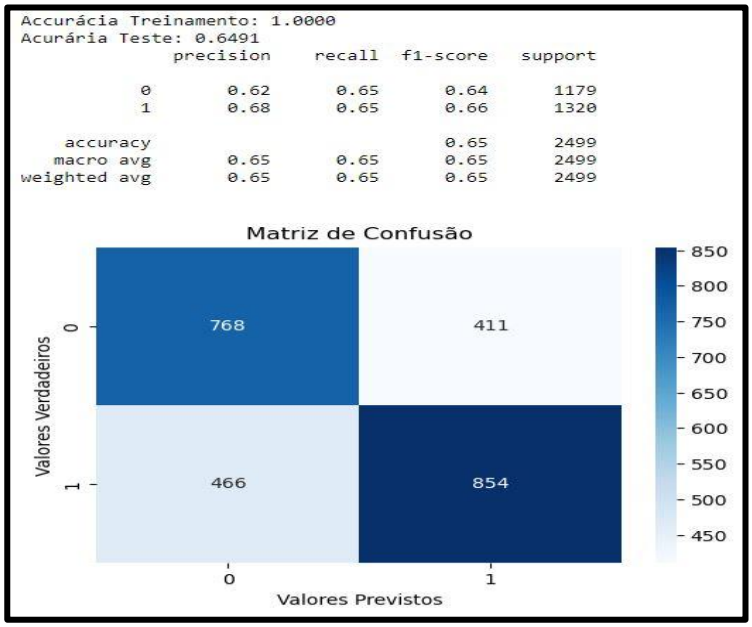
O algoritmo de Random Florest classificou corretamente 100% dos valores em sua fase de teste, contudo analisando a acurácia do modelo na fase de teste obteve um total de 70%. O *recall e o f1-score* para a classe 0 teve resultados abaixo dos 70%, o

que mostra que este algoritmo também obteve melhor resultados para a classificação dos times da cor azul.

A matriz de confusão para este segundo algoritmo classificou corretamente o time vermelho um total de 806 partidas, além de 955 partidas corretamente para o time azul. Em comparação com o algoritmo de regressão logística, observa-se maior números de falsos positivos, sendo 391 para o time vermelho e 347 para o time azul, respectivamente.

Por fim, o último algoritmo analisado foi o de árvore de decisão. A Figura 33 mostra os resultados obtidos para este modelo:

Figura 33 – Resultados obtidos através da Árvore de Decisão para o primeiro modelo



Fonte: autoria própria.

Assim como no modelo de Random Florest a fase de treinamento do modelo de árvore de decisão teve um total de 100% de acertos, contudo na fase de teste a árvore de decisão conseguiu predizer apenas 64,91% dos valores corretamente. Além disso, dentre todos os algoritmos propostos, este foi o que mais encontrou dificuldades para predizer as classes que referência o time vermelho, possuindo apenas 62% de precisão contra 68% do time azul.

A matriz de confusão para este terceiro algoritmo classificou corretamente o time vermelho um total de apenas 768 partidas, além de 854 partidas corretamente para o time azul. Em comparação com os modelos acima, observa-se maior números de falsos

positivos entre todos os modelos, sendo 411 para o time vermelho e 466 para o time azul respectivamente.

Outro fator que é importante analisar dentro de algoritmos de *machine learning* é entender o que cada variável representa de importância para aquele algoritmo. A Tabela 6 representa a importância das variáveis para este primeiro modelo contendo um total de 7 variáveis que predizem a variável Blue_result para cada algoritmo de regressão logística, random florest e árvore de decisão, respectivamente:

Tabela 6 – Importância das variáveis para cada algoritmo do primeiro modelo

Regressão Logística			**Random Florest**			**Árvore de Decisão**	
Variáveis	**Coeficiente**		**Variável**	**Importância**		**Variável**	**Importância**
Red_deathsat15	2,501017		*Red_xpdiffat15*	0,248380		*Red_xpdiffatt15*	0,364877
Blue_xpat15	1,133942		*Blue_xpat15*	0,184991		*Red_csdiffat15*	0,160061
Blue_firsttower	0,702201		*Red_xpat15*	0,178305		*Blue_xpat15*	0,153447
Red_xpat15	-1,414179		*Red_csdiffat15*	0,165762		*Red_xpat15*	0,148428
Red_xpdiffat15	-2,280908		*Red_killat15*	0,081443		*Red_killsat15*	0,070491
Red_killat15	-2,521102		*Red_deathsat15*	0,079661		*Red_deathsat15*	0,063266
Red_csdiffat15	-3,571518		*Blue_firsttower*	0,061458		*Blue_firsttower*	0,039430

Fonte: autoria própria.

Para o modelo de regressão logística foi utilizado o coeficiente estimado. Este tipo de análise indica a magnitude e direção entre a variável preditora e as variáveis de respostas. Através do *log-odds* é possível indicar alteração de coeficiente, por exemplo, a variável *Red_deathsat15* tem um aumento unitário positivo associado a um aumento de 2.501017 para a variável de resposta 1.

Essa relação positiva indica que o aumento da variável preditora está associado a um aumento de ocorrência de um evento acontecer. Em síntese, os coeficientes que possuem valores positivos estão relacionados diretamente ao aumento da classe 1, ou seja, do time azul. Da mesma forma, variáveis que possuem coeficientes negativos estão relacionando diretamente ao aumento da classe 0, ou seja, o time vermelho.

Para a verificação tanto do *random florest* como da árvore de decisão foi utilizado o feature_importances_ do scikit-learn. Este método classifica as variáveis mais

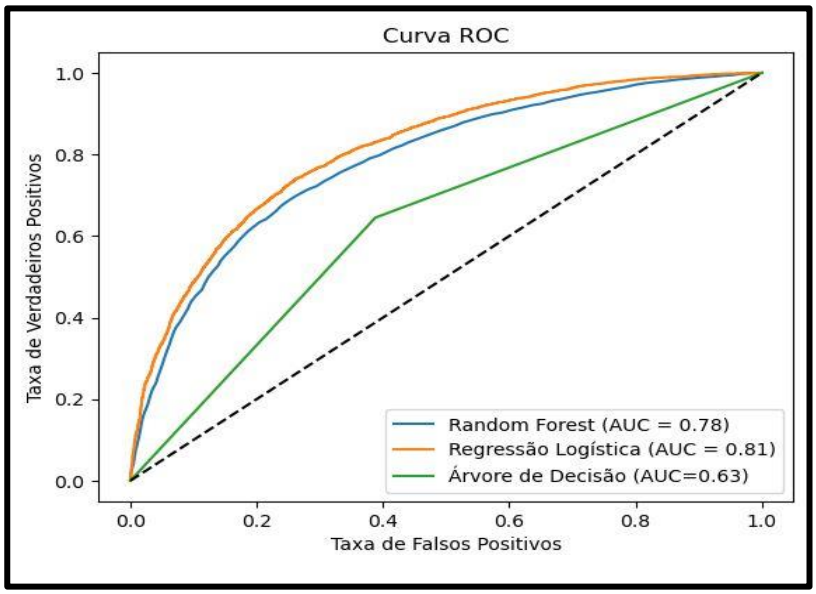
importantes para cada modelo. Por exemplo, para o algoritmo de random florest a variável que teve o maior peso de importância no modelo foi *Red_xpdiffat15, Blue_xpat15, Red_xpat15* e assim por diante. Já na árvore de decisão as principais variáveis deste modelo foram: *Red_xpdiffatt15, Red_csdiffat15, Blue_xpat15* e assim por diante.

Como pode-se observar, apesar dos modelos de *random florest* e árvore de decisão serem relacionados, cada um deles relaciona as variáveis e sua importância totalmente diferentes. Por exemplo, a segunda variável de maior importância para a árvore de decisão é *Red_csdiffat15,* enquanto no *random florest* a segunda variável é *Blue_xpat15.*

Por fim, outro meio de analisar os diagnósticos e desempenhos dos modelos é através da curva de *Receiver Operating Characteristc* (ROC). A curva de ROC exemplifica o quão bem um modelo consegue se diferenciar duas classificações. No caso deste trabalho, distinguir a vitória ou a derrota de uma equipe no *League of Legends.*

A ROC possui dois parâmetros, que são a taxa de verdadeiros positivos e a taxa de falsos positivos. Outra métrica utilizada para simplificar a curva de ROC é através da *Area Under the ROC Curve* (AUC), que tem como objetivo resumir a ROC em valores que variam de 0 até 1. Quanto maior o valor de AUC, melhor a capacidade de classificação do algoritmo. A Figura 34 apresenta a curva de ROC para os algoritmos de regressão logística, random florest e árvore de decisão.

Figura 34 – Curva ROC do primeiro modelo



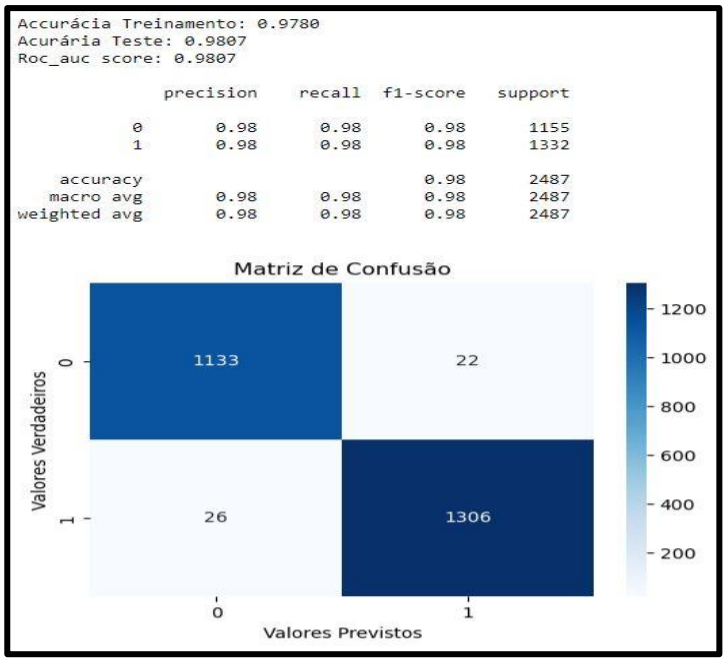
Fonte: autoria própria.

Dentre os algoritmos propostos neste trabalho, para a observação dos 15 minutos iniciais de partida do *League of Legends* para predizer vitórias e derrotas de uma equipe, observou-se que o melhor algoritmo foi o de regressão logística, contendo uma AUC de 0.81, seguido por *random florest* com AUC de 0.78. Por fim, a árvore de decisão, com uma AUC de 0.63.

5.3 Resultados e Análises do Segundo Modelo

O segundo modelo conta com 9.947 registros e um total de 11 variáveis preditoras para determinar a vitória ou derrota de uma equipe utilizando dados do final da partida. A Figura 36 mostra os resultados obtidos para o algoritmo de Regressão Logística.

Figura 35 – Resultados obtidos da Regressão Logística para o segundo modelo



Fonte: autoria própria.

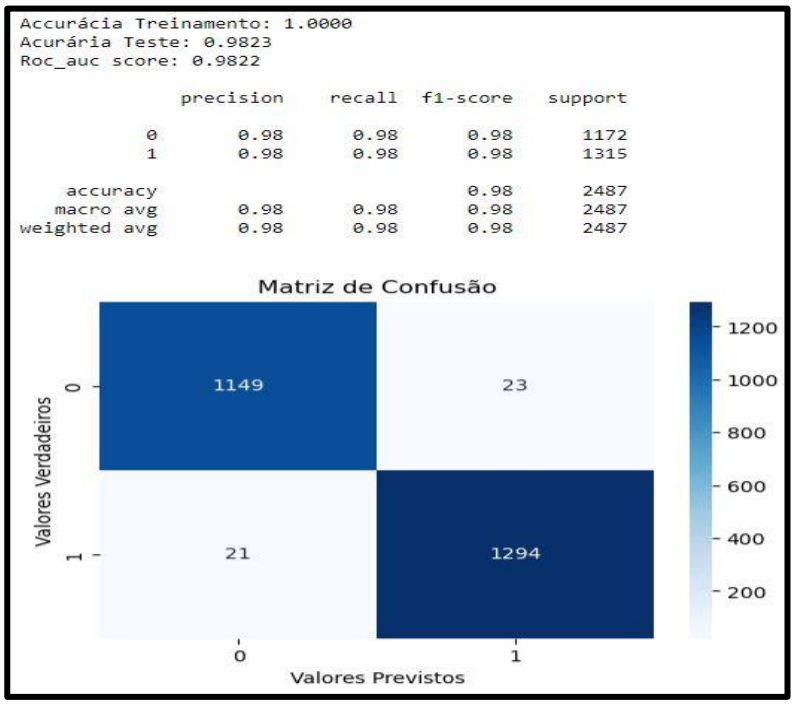
Para o segundo modelo a regressão logística classificou corretamente em sua fase de treinamento um total de 97,8%, contudo na fase de teste teve um pouco uma pequena melhora para 98,07% de classes classificadas corretamente. Observando os valores de *precision, recall e f1-score* é possível identificar que este modelo conseguiu classificar igualmente as classes 0 e 1, não tendo discrepância dos dados conforme o primeiro modelo.

A matriz de confusão do segundo modelo utilizando regressão logística classificou corretamente o time vermelho um total 1133 partidas, além de 1306 partidas

corretamente para o time azul. Observa-se que existe poucos falsos positivos entre as classes, o que indica que este modelo classificou acima da média, sendo apenas 22 para o time vermelho e 26 para o time azul respectivamente.

O segundo algoritmo analisado para este modelo foi o de *Random Florest*. A Figura 36 mostra os resultados obtidos para este modelo:

Figura 36 – Resultados obtidos do *Random Florest* para o segundo modelo



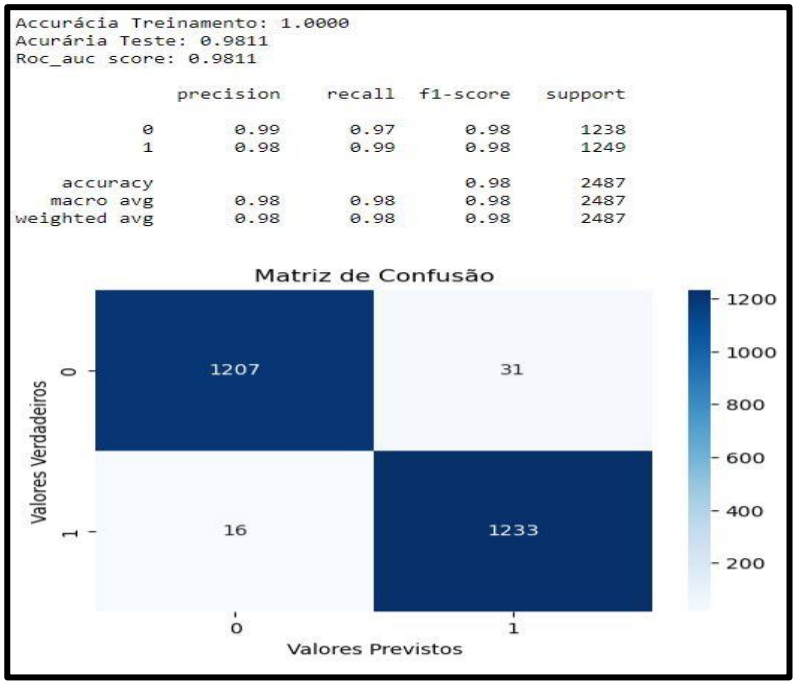
Fonte: autoria própria.

Assim como no modelo de 15 primeiros minutos o algoritmo de *random florest* classificou corretamente 100% dos valores na fase de teste, contudo, diferentemente no primeiro neste modelo a fase de teste para este segundo modelo obteve um total de 98,23% de taxa de acertos. Assim como no algoritmo de regressão logística, ambas as classes conseguiram classificar igualmente.

A matriz de confusão para este segundo algoritmo classificou corretamente o time vermelho um total de 1149 partidas, além de 1294 partidas corretamente para o time azul. Em comparação com o algoritmo de regressão logística, observa-se baixos falsos positivos, sendo 23 para o time vermelho e 21 para o time azul respectivamente.

Por fim, o último algoritmo analisado do segundo foi o de árvore de decisão. A Figura 37 mostra os resultados obtidos para este modelo:

Figura 37 – Resultados obtidos através da Árvore de Decisão para o segundo modelo.



Fonte: autoria própria.

Utilizando a árvore de decisão para o segundo modelo obteve-se um total de 100% de acertos. Em contrapartida, na fase de teste, diferentemente do primeiro modelo de análise dos 15 primeiros minutos de partida, houve uma grande taxa de acertos sendo de 98,11% dos valores foram classificados corretamente. A matriz de confusão para este terceiro algoritmo classificou corretamente o time vermelho com um total de apenas 1207 partidas, além de 1233 partidas corretamente para o time azul. Também houve baixos números para falsos positivos, sendo 31 para o time vermelho e 16 para o time azul, respectivamente.

Para este segundo modelo contendo um total de 10 variáveis que predizem a variável Blue_result, a Tabela 7 representa a importância dessas variáveis para os modelos de regressão logística, *random florest* e árvore de decisão respectivamente:

Tabela 7 – Importância das variáveis para cada algoritmo do segundo modelo

Regressão Logística			**Random Florest**			**Árvore de Decisão**	
Variáveis	**Coeficiente**		**Variável**	**Importância**		**Variável**	**Importância**
Blue_inhibitors	7,211546		*Blue_inhibitors*	0,313702		*Blue_inhibitors*	0,862804
Red_deaths	5,982353		*Red_inhibitors*	0,312839		*Red_inhibitors*	0,066093
Blue_doublekills	2,401271		*Red_earnedgold*	0,149776		*Red_earnedgold*	0,028145
Blue_barons	1,482342		*Red_barons*	0,10249		*Red_deaths*	0,015494
Red_barons	0,069977		*Red_deaths*	0,046715		*Red_dragons*	0,006986
Blue_frsttothreetowers	0,027079		*Red_doublekills*	0,023865		*Red_doublekills*	0,006260
Red_dragons	-2,865474		*Blue_barons*	0,018474		*Blue_doublekills*	0,004521
Red_doublekills	-3,819479		*Blue_doublekills*	0,017629		*Red_barons*	0,003972
Red_inhibitors	-7,27114		*Red_dragons*	0,011533		*Blue_frsttothreetowers*	0,003128
Red_earnedgold	-10,72253		*Blue_frsttothreetowers*	0,002975		*Blue_barons*	0,002598

Fonte: autoria própria

Utilizando o coeficiente estimado da regressão logística é possível notar que as variáveis: *Blue_inhibitors, Red_deaths, Blue_doublekills, Blue_barons, Red_barons* tem uma relação positiva para a classe preditora 1, ou seja, essas variáveis favorecem para que este evento time azul ganhar. Por outro lado, as variáveis: *Red_dragons, Red_doublekills, Red_inhibitors e Red_earnedgold* possuem uma relação negativa com a classe preditora 1, ou seja, essas variáveis não favorecem para o evento time azul, portanto, elas favorecem ao time vermelho vencer.

No contexto do *League of Legends,* quando ocorre a destruição de inibidores por parte do time azul, quem será beneficiado será propriamente o time azul. Desta forma, essa variável correlaciona positivamente para a vitória. Da mesma forma, a variável *Red_earnedgold* correlaciona à quantidade ouro que o time vermelho obtém. Sendo assim, menores são as chances de o time azul conseguir a vitória.

Para a verificação tanto do random florest como da árvore de decisão foi utilizado o feature_importances_ do scikit-learn. Este método classifica as variáveis mais importante para cada modelo.

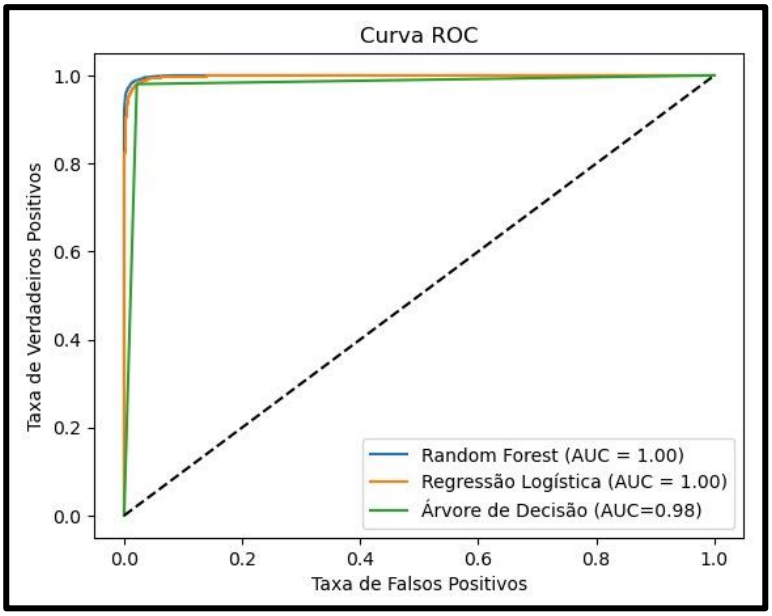
Pode-se notar que o topo de grau de importância por parte da árvore de decisão é referente à variável Blue_inhibitors, porém, analisando mais detalhadamente pode-se notar, através da Tabela 7, que as demais variáveis tem grande discrepância de importância quando comparado à primeira variável de importância.

Já no modelo de random florest, por conta desse algoritmo criar diversas árvores de decisão randômicas para um mesmo problema e analisando quais dessas variáveis tiveram um melhor desempenho, pode-se notar que há um certo equilíbrio entre as variáveis mais importante para este algoritmo, desta forma, o modelo de random florest é mais confiável do que o modelo de árvore de decisão.

O *League of Legends* apesar de ser um jogo no qual qualquer mínima vantagem de um time pode ocasionar a derrota de outro time, contudo, existe um certo balanceamento no jogo que indica que mesmo a destruição do inibidor inimigo ou não, não garante 100% das vitórias de uma equipe. Por exemplo, dentro de uma partida de *League of Legends* a cada destruição de um inibidor, começa uma contagem de *respawn* para que este inibidor ressurge depois dos 3 minutos da destruição.

Supondo que um time azul destruiu 3 inibidores, porém não conseguiu ganhar a partida e o time vermelho conseguiu maior quantidade de ouro e necessitou apenas destruir um 1 inibidor para levar o *nexus.* Isso garante que apesar dessa variável ser importante, existem outras variáveis que também tem um bom grau de importância para conseguir a vitória. Finalmente a Figura 38 mostra o desempenho da curva de ROC e da AUC variando de 0 até 1 para os algoritmos de regressão logística, *random florest* e árvore de decisão para este experimento:

Figura 38 – Curva ROC do segundo modelo



Fonte: autoria própria.

Os resultados obtidos e apresentados na Figura 38 mostram que para os modelos de *random florest* e regressão logística obteve-se uma AUC de 1.00. A árvore de decisão teve uma AUC de 0.98, contudo, todos os resultados para este modelo de coleta dos dados ao final do jogo tiveram uma boa performance.

6 CONSIDERAÇÕES FINAIS

Este trabalho teve como objetivo de responder a seguinte questão de pesquisa: - **Quais fatores influenciam na classificação dos resultados de uma partida baseado no desempenho das equipes no contexto do jogo *League of Legends*, em dois momentos distintos do jogo?**

Os resultados obtidos atenderam aos objetivos do trabalho. Inicialmente são criados dois modelos de estudos, um que tenta predizer a vitória coletando dados aos 15 minutos de partida e o outro que coleta dados ao final das partidas do *League of Legends.* Desta forma, os dados são adequados para que produzam resultados das equipes, em seguida aplicou-se os conceitos e técnicas de mineração de dados utilizando critérios de seleção de variáveis: seleção de variáveis, variáveis mais importantes, limpeza dos dados, análise de redundâncias. Por fim, é aplicados os algoritmos de *machine learning* para os modelos e avaliado qual obteve melhor desempenho.

A aplicação da mineração de dados dentro dos processos de descoberta de conhecimento em base de dados se mostrou uma grande aliada na tomada de decisão de vitórias ou derrotas em partidas do *League of Legends.* A aplicação dessas técnicas mostra que, é possível prever bons graus de acurácia para ambos os modelos.

A análise dos resultados obtidos nos experimentos com o Python, foi feita por meio de uma análise descritivas dos dados, em que são apresentados a capacidade preditiva proveniente da comparação dos resultados dos algoritmos de *machine learning:* regressão logística, *random florest* e árvore de decisão para os dois modelos de análise. Além disso, este trabalho procurou entender quais fatores influenciam na vitória, analisando o grau de importância das variáveis para cada algoritmo em relação aos modelos de análise aos 15 minutos e ao final de partida.

Este estudo permitiu concluir que a aplicabilidade da mineração de dados no mundo do *League of Legends* é possível, entretanto, existem diversos estudos possíveis no jogo. Para este trabalho, foi analisado dados aos 15 minutos da partida e ao final de jogo utilizando os algoritmos propostos a escolha desse estágio de jogo se deu por ser o final do *early game* e o início do *mid game*. A análise dos 15 primeiros minutos

constatou que a utilização da regressão logística e *random florest* tiveram melhores resultados.

Por outro lado, analisando o modelo de dados ao final de jogo, todos os algoritmos obtiveram bons resultados acima de 98% de acurácia dentro desta análise. As partidas que os algoritmos classificação incorretamente para este modelo são partidas no qual o time A tem vantagem mesmo após o final da partida. Entretanto, o time B foi o vencedor. Dentro das partidas profissionais, pode-se notar que poucas vezes isso aconteceu dentro do modelo de predição, no qual foi possível concluir que, para este conjunto de dados de partidas profissionais, o time que tem a vantagem quase sempre sairá com a vitória.

Para continuidade desta pesquisa, sugere-se os seguintes trabalhos futuros:

- Incluir grau de importância das variáveis dentro do contexto de predição.
- Generalizar a base de dados para partidas casuais e competitivas, analisando suas diferenças.
- Realizar teste em dados de *mid* e *late game*, avaliando o grau de acertos dos modelos preditivos para essas fases de jogo.
- Correlacionar a escolha do campeão com o desempenho esperado nos três estágios de jogo: *early*, *mid* e *late game*.

REFERÊNCIAS

BAIA, C. ***Decision Tree e Random Forest****.* 2016. Disponível em: http://carlosbaia.com /2016/12/24/decision-tree-e-random-forest/. Acesso em: 10 novembro 2022.

BATISTA, G. E. A. P. A. **Pré-processamento de Dados em Aprendizado de Máquina Supervisionado**. 2003. Tese (Doutorado) - Curso de Ciências - Ciências de Computação e Matemática Computacional, Universidade de São Paulo, São Carlos, 2003.

BARCELLOS, R. L. **Suporte à Tomada de Decisão Estratégica no Âmbito de Esports: o caso do *League of Legends*.** 2017. Dissertação (Mestrado em Engenharia) – Programa de Pós-Graduação em Engenharia de Produção, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2017.

BORISOV, A. ***Most Watched Esports Tournaments of 2021***. 2021. Disponível em: https://escharts.com/news/most-watched-tournaments-2021. Acesso em: 10 outubro 2022.

BTOBET, ***Brazil Betting Focus***. 2022. Disponível em: https://www.btobet.com /downloads/brazil-betting-focus-industry-report-2/?lang=pt-br. Acesso em: 05 outubro 2022.

CABENA P. et al. ***Discovering data mining: from concept to implementation****. Englewood Cliffs: Prentice Hall,* 1998.

CANOSSA, A.; DRACHEN, A.; EL-NASR, M. S. ***Game analytics.*** *2016. Springer*, 2016.

CARBONE, F. **LoL: como funciona o sistema de *rankeds***. 2021. Disponível em: https://ge.globo.com/esports/lol/zantins/noticia/lol-como-funciona-o-sistema-de-rankeds.ghtml. Acesso em: 02 novembro 2022.

CASTRO, L. N. de; FERRARI, D. G. **Introdução à Mineração de Dados: conceitos básicos, algoritmos e aplicações**. São Paulo: Saraiva, 2016.

CUTLER, A; BREIMAN, L.; ***Random Forests****.* 2001. Disponível em: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm. Acesso em: 10 novembro 2022.

DAMÁSIO, B. F. **O que é Correlação de Pearson?.** 2021. Disponível em: https://psicometriaonline.com.br/o-que-e-correlacao-de-pearson/. Acesso em: 22 junho 2023.

DANTAS, E. R. G.; et al. **O Uso da Descoberta de Conhecimento em Base de Dados para Apoiar a Tomada de Decisões**. Centro Universitário de João Pessoa – UNIPÊ. Centro de Informática – Universidade Federal de Pernambuco, Pernambuco, 2008.

DRACHEN, A.; et al. ***Skill-Based Differences in Spatio-Temporal Team Behaviour in Defence of The Ancients 2 (DotA 2)****. In Games Media Entertainment (GEM),* 2014 IEEE. IEEE, 2014.

ESPORTS CHARTS. ***Top esports games in 2021 by prize Money****. 2021 Disponível em: https://escharts.com/top-games?year=2021. Acesso em: 02 novembro 2022.*

FAYYAD, U. M. et al. ***Advances in Knowledge Discovery and Data Mining****. 1996. Cambridge: AAAI Press/MIT Press, California*, 1996.

FERREIRA, R. S. **10 ferramentas e bibliotecas para trabalhar com *data mining* e *Big data*.** 2017. Disponível em: https://imasters.com.br/data/10-ferramentas-e-bibliotecas-para-trabalhar-com-data-mining-e-big-data-parte-01. Acesso em: 10 novembro 2022.

FLOTER, R. **Os melhores MOBAs para Android e iOS. 2021**. Disponível em: https://canaltech.com.br/jogos-mobile/os-melhores-mobas-para-android-e-ios-117694/. Acesso em: 05 novembro 2022.

FRANK, E.; HALL, M. A. & WITTEN, I. H. ***The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques****.* [S.l.]: Morgan Kaufmann, 2016.

GRANDE PRÊMEIO. **O crescimento exponencial de apostas em eSports**. 2022. Disponível em: https://www.grandepremio.com.br/esports/noticias/o-crescimento-exponencial-de-apostas-em-esports/. Acesso em: 05 novembro 2022.

GE. **Mercado de eSports: faturamento, audiência e o cenário no Brasil**. 2021. Disponível em: https://ge.globo.com/sc/noticia/o-mercado-de-esports-faturamento-audiencia-e-o-cenario-no-brasil.ghtml. Acesso em: 05 novembro 2022.

GIL, A. C. **Como Elaborar Projetos de Pesquisa**. 6. ed. São Paulo: Editora Atlas Ltda., 2017.

GOLDSCHMIDT, R.; PASSOS, E. ***Data Mining*** **um Guia Prático. Conceitos, Técnicas, Ferramentas, Orientações e Aplicações**. Ed. Campus, Rio de Janeiro, 2005.

GOMES, L. **Conheça os melhores MOBAs de 2021**. 2022. Disponível em: https://www.showmetech.com.br/melhores-mobas-de-2021/. Acesso em: 05 novembro 2022.

HAN, J.; KAMBER, M. ***Data Mining: Concepts and Techniques****. Morgan Kaufmann Publishers, 2001.*

HELLEU, B.; et al. ***Mining Tracks of Competitive Video Games****. AASRI Procedia*, 8:82-87, 2014.

HITAR-GARCÍA, J. A.; MORÁN-FERÁNDEZ, L.; BOLÓN-CANEDO, V. ***Machine Learning Methods for Predicting League of Legends Game Outcome.*** 2021. Disponível: https://ieeexplore.ieee.org/document/9720122. Acesso em: 18 maio 2023.

HO, T. K. ***Random Decision Forests****. AT&T Bell Laboratories,* 2001.

JUPYTER. ***The Jupyter Notebook***. 2022. Disponível em: https://jupyter-notebook.readthedocs.io/en/stable/. Acesso em: 10 novembro 2022.

KINKADE, Nicholas; LIM, Kyung yul Kevin. 2015. DOTA 2 Win Prediction. Disponível em: https://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/018.pdf. Acesso em: 20 junho 2023.

LEAGUE OF LEGENDS. **Boas-Vindas ao *Rift* Aprenda o Básico**. 2022. Disponível em: https://www.leagueoflegends.com/pt-br/how-to-play. Acesso em: 10 outubro 2022.

LUCCA, M. ***League of Legends*** **tem quantos jogadores ativos regularmente em 2022?**. 2022. Disponível em: https://br.millenium.gg/noticias/11574.html#:~:text=Considerando%20os%20dados%20recentes%20do,que%20jogaram%20regularmente%20nesse%20per%C3%ADodo. Acesso em: 03 outubro 2022.

LUNELLI, L. M. **Previsão de resultado de jogos da NBA com algoritmos de *machine learning***. Dissertação (Mestrado) — Universidade Nova de Lisboa, Lisboa, 2020.

MARQUES, R. **LoL: Pico diário de jogadores é de 8 milhões, diz *Riot***. 2019. Disponível em: https://maisesports.com.br/lol-pico-8-milhoes. Acesso em: 05 outubro 2022.

MATPLOTLIB. ***Matplotlib*: *3.6.2 documentation***. 2022. Disponível em: https://matplotlib.org/stable/index.html. Acesso em: 10 novembro 2022.

MESQUITA, P. S. B. **Um Modelo de Regressão Logística para Avaliação dos Programas de Pós–Graduação no Brasil**. Dissertação (Mestrado) — Universidade Estadual do Norte Fluminense, Campo dos Goytacazes, 2014.

MEZZOMO, G. **Consumo e Fandom em jogos: um estudo sobre *League of Legends.*** Santa Maria: UFSM, 2014. Monografia (Graduação) – Programa de graduação em Ciências Sociais e Humanas, Universidade Federal de Santa Maria, Rio Grande do Sul, 2014.

MIGUEL, T. **Regressão logístic**a. 2020. Disponível em: https://aprenderdatascience .com/regressao-logistica/. Acesso em: 05 novembro 2022.

MILELLA, V. ***League of Legends rank distribution in solo queue – november 2022***. 2022. Disponível em: https://www.esportstales.com/league-of-legends/rank-distribution-percentage-of-players-by-tier. Acesso em: 02 novembro 2022.

MINOTTI, M. ***The History of Mobas: From mod to sensation***. 2014. Disponível em: https://venturebeat.com/games/the-history-of-mobas-from-mod-to-sensation/view-all/. Acesso em: 04 novembro 2022.

MONTGOMERY, D. C. **Introdução ao Controle Estatístico da Qualidade**. Trad. Ana Maria Lima de Farias, Vera Regina Lima de Farias e Flores; 4. ed. [Reimpr.]. Rio de Janeiro: LTC, 2014.

MOURA, K. **Ciclo de vida dos dados: kdd process**. 2019. Disponível em: https://medium.com/@kvmoura/kdd-process-9b8e3062142. Acesso em: 20 outubro 2022.

MOURÃO, R. N. **Mineração de Dados para Previsão de Renda de Clientes com Contas-Correntes Digitais**. 2018. 70 f. Dissertação (Mestrado) - Curso de Computação Aplicada, Universidade de Brasília, Brasília, 2018.

NASCIMENTO Jr., F. **Caracterização de Perfis de Comportamento de Equipes em *League of Legends*.** Campina Grande: UFCG, 2017. Tese (Mestrado) – Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Campina Grande, Campina Grande, 2017.

NEVES, B. O. **Mineração de Dados Aplicada à Predição de Resultados de jogos de Basquete**. Ouro Preto: UFOP, 2022. Monografia (Graduação) – Instituto de Ciências Exatas e Biológicas Departamento de Computação, Universidade Federal de Ouro Preto, Ouro Preto, 2022.

NOVAIS, G.; TARTAGLIA, R. **LoL: skins, runas, personagens; o que é e tudo sobre o MOBA da *Riot***. 2021. Disponível em: https://ge.globo.com/esports/lol/noticia/lol-skins-runas-personagens-o-que-e-e-tudo-sobre-o-moba-da-riot-games.ghtml. Acesso em: 02 novembro 2022.

NUZZO, R. L. ***Histograms: A useful data analysis visualization.*** 2019. Disponível em: https://onlinelibrary.wiley.com/doi/full/10.1002/pmrj.12145. Acesso em: 21 junho 2023.

PACETE, L. G. **2022 promissor: mercado de games ultrapassará US$ 200bi até 2023.** 2022. Disponível em: https://forbes.com.br/forbes-tech/2022/01/com-2022-decisivo-mercado-de-games-ultrapassara-us-200-bi-ate-2023. Acesso em: 11 outubro 2022.

PADHY, N., MISHRA, P., PANIGRAHI, R. ***The Survey of Data Mining Applications and Feature Scope****. International Journal of Computer Science, Engineering and Information Technology, 2012.*

PANDAS. ***Pandas documentation***. 2022. Disponível em: https://pandas.pydata.org/docs/. Acesso em: 10 novembro 2022.

PEREIRA, W. ***Worlds 2022: fase de grupos teve queda de quase 40% na audiência.*** *2022.* Disponível em: https://ge.globo.com/esports/lol/noticia/2022/10/20/worlds-2022-

fase-de-grupos-teve-queda-de-quase-40percent-na-audiencia.ghtml. Acesso em: 05 novembro 2022.

PINHEIRO, C. A. R. **Inteligência analítica.** Ciência Moderna, 2008.

RELICH, M.; MUSZYNSKI, W. ***The use of intelligent systems for planning and scheduling of product development projects***. Procedia Computer Science v. 35, 2014.

ROSA, C.R.M. **Uma metodologia para a descoberta de conhecimento em bases de dados visando a classificação de padrões**. Curitiba: PUCPR, 2017. Tese (Doutorado) – Programa de Pós-Graduação em Engenharia de Produção e Sistemas, Pontifícia Universidade Católica do Paraná, Curitiba, 2017.

SCOLA, A. ***League of Legends*: guia para iniciantes**. 2022. Disponível em: https://tecmasters.com.br/league-of-legends-guia-para-iniciantes/. Acesso em: 02 novembro 2022.

SCIKIT-LEARN. ***User Guide***. 2022. Disponível em: https://scikit-learn.org/stable/user_guide.html. Acesso em: 10 novembro 2022.

SCIKIT-LEARN DEVELOPERS. **1.13. *Feature selection*.** 2023. Disponível em: https://scikit-learn.org/stable/modules/feature_selection.html. Acesso em: 21 junho 2023.

SHEN, Q. ***A Machine Learning Approach to predict the Result of League of Legends.*** 2022. Disponível em: https://ieeexplore.ieee.org/document/9763608. Acesso em: 23 junho 2023.

SILVA, L. A.; PERES, S. M.; BOSCARIOLI, C. **Introdução à mineração de dados: com aplicações em r**. Rio de Janeiro: Elsevier, 2016.

SILVA, V. N. **Desenvolvimento de agentes inteligentes para jogos Moba.** Belo Horizonte: UFMG, 2016. Tese (Mestrado) – Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, 2016.

SOUZA, R. T. **Aplicação de algoritmos classificadores para previsão de vitória em uma partida de *League of Legends*.** Porto Alegre: UFRGS, 2017. Monografia (Graduação) – Programa de Graduação em Engenharia de Produção, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2017.

STEINBACH, M.; TAN, P. N.; KUMAR, V. **Introdução ao *datamining*: mineração de dados**. Rio de Janeiro: Ciência Moderna, 2009.

TRISTÃO, H. ***The Esport Audience Will Pass Half a Bilion in 2022 as Revenues, Engagement, & New Segments Flourish.*** 2022. Disponível em: https://newzoo.com/insights/articles/the-esports-audience-will-pass-half-a-billion-in-2022-as-revenue-engagement-esport-industry-growth. Acesso em: 15 outubro 2022.

VASQUEZ, A. **Final do CBLOL 2022 tem pico com 331 mil espectadores.** 2022. Disponível em: https://ge.globo.com/esports/lol/noticia/2022/09/05/cblol-final-tem-pico-com-331-mil-espectadores.ghtml. Acesso em: 05 novembro 2022.

VICTOR, M. **Como os eSports se tornaram a terceira categoria de apostas mais popular no Brasil**. 2022. Disponível em: https://www.jornadageek.com.br/os-esports-se-tornaram-a-terceira-maior-categoria-de-apostas-no-brasil/. Acesso em: 05 novembro de 2022.

WAZLAWICK, R. S. **Metodologia da Pesquisa para Ciência da Computação**. 2ª. ed. [S.l.]: Campus, 2014.

APÊNDICE A - VARIÁVEIS REDUNDANTES E NÃO DESEJADAS REMOVIDAS DO *DATASET*

Quadro 4 – Variáveis redundantes e não desejadas removidas do *Dataset*

Variáveis	Descrição
Teamkills	O número total de abates realizados pelo time durante a partida.
teamdeaths	O número total de mortes sofridas pelo time durante a partida.
team kpm	*Kills* por minuto do time.
*ckpm**	Kills e assistências por minuto. Sua fórmula é igual (*kills* + *assists*) / tempo de jogo.
opp_dragons	O número total de dragões capturados pelos oponentes durante a partida.
*elementaldrakes**	O número total de dragões elementais capturados pelo time durante a partida
*opp_elementaldrakes**	O número total de dragões elementais capturados pelos oponentes durante a partida.
*infernals**	O número total de dragões elementais de fogo capturados pelo time durante a partida.
*mountains**	O número total de dragões elementais de montanha capturados pelo time durante a partida.
*clouds**	O número total de dragões elementais de nuvem capturados pelo time durante a partida.
*oceans**	O número total de dragões elementais de oceano capturados pelo time durante a partida.
*chemtechs**	O número total de dragões elementais de química capturados pelo time durante a partida.
*hextechs**	O número total de dragões elementais de *hextec* capturados pelo time durante a partida.
*elders**	Quantidade dragões ancestrais capturados pelo time durante a partida.
opp_elders	Total de dragões ancestrais capturados pelos oponentes durante a partida. Aparece depois de 35 minutos de partida.
opp_heralds	O número total de Arautos capturados pelos oponentes durante a partida.
opp_barons	O número total de Barões Nashor capturados pelo time inimigo durante a partida.
opp_towers	O número total de torres destruídas pelos oponentes durante a partida.

*turretplates**	Total de placas de torres que foram destruídas pelo time durante a partida.
opp_turretplates	Total de placas de torres que foram destruídas pelos oponentes durante a partida.
opp_inhibitors	Total de inibidores destruídos pelos oponentes durante a partida.
Dpm	Dano por minuto.
*damagetakenperminute**	Dano recebido por minuto pelo time durante a partida
*damagemitigatedperminute**	Dano mitigado por minuto pelo time durante a partida.
Wpm	Quantidade de sentinelas colocadas durante uma partida por minuto.
Wcpm	Quantidade de sentinelas destruídas durante uma partida por minuto.
*controlwardsbought**	O número total de sentinelas de controle compradas pelo time durante a partida.
Vspm	Visão de sentinelas do time por minuto
earned gpm	Ouro ganho pelo time por minuto.
*goldspent**	Ouro gasto pelo time durante a partida.
*gspd**	Diferença de ouro por minuto entre os times.
*minionkills**	O número total de tropas mortas pelo time durante a partida.
*monsterkills**	Número total de monstros mortas pelo time durante a partida.
*goldat10**	Total de ouro obtido pela equipe aos 10 minutos de partida
*xpat10**	Experiência total ganha pelo time aos 10 minutos de partida.
*csat10**	O número total de tropas mortas pelo time aos 10 minutos de partida.
opp_goldat10	Ouro total ganho pelo time adversário aos 10 minutos de partida.
opp_xpat10	Experiência total ganha pelo time adversário aos 10 minutos de partida.
opp_csat10	O número total de tropas mortas pelo time adversário aos 10 minutos de partida.

*golddiffat10**	Diferença de ouro entre os times aos 10 minutos de partida.
*xpdiffat10**	Diferença de experiência entre os times aos 10 minutos de partida.
*csdiffat10**	Diferença no número total de tropas mortas pelo jogador e seu oponente de rota aos 10 minutos de partida.
*killsat10**	Quantidade de abates pela equipe aos 10 minutos de partida.
assistsat10*	O número total de assistências concedidas pelo time aliado aos 10 minutos de partida.
deathsat10*	Total de mortes do time aos 10 minutos de partida.
opp_killsat10	Número total de abates do time adversário aos 10 minutos de partida.
opp_assistsat10	Número total de assistências concedidas pelo time adversário aos 10 minutos de partida
opp_deathsat10	Número total de mortes concedidas pelo time adversário aos 10 minutos de partida

Fonte: autoria própria.