



**UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO**  
**CENTRO TECNOLÓGICO**  
**COLEGIADO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO**

Lucas Quintino Frinhani

# **Uso de Aprendizado de Máquina para Predição de Resultados no Mercado de Apostas**

Vitória, ES

2021

Lucas Quintino Frinhani

# **Uso de Aprendizado de Máquina para Predição de Resultados no Mercado de Apostas**

Monografia apresentada ao Curso de Ciência da Computação do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Colegiado do Curso de Ciência da Computação

Orientador: Prof. Flávio Miguel Varejão

Vitória, ES

2021

---

Lucas Quintino Frinhani

Uso de Aprendizado de Máquina para Predição de Resultados no Mercado de Apostas/ Lucas Quintino Frinhani. – Vitória, ES, 2021-

67

Orientador: Prof. Flávio Miguel Varejão

Monografia (PG) – Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Colegiado do Curso de Ciência da Computação, 2021.

I. Varejão, Flávio Miguel. II. Universidade Federal do Espírito Santo. IV. Uso de Aprendizado de Máquina para Predição de Resultados no Mercado de Apostas

CDU 02:141:005.7

---

Lucas Quintino Frinhani

# **Uso de Aprendizado de Máquina para Predição de Resultados no Mercado de Apostas**

Monografia apresentada ao Curso de Ciência da Computação do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Trabalho aprovado. Vitória, ES, 07 de Outubro de 2021:

---

**Prof. Flávio Miguel Varejão**  
Orientador

---

**Prof. Thomas Walter Rauber**  
Universidade Federal do Espírito Santo

---

**Prof. Thiago Oliveira dos Santos**  
Universidade Federal do Espírito Santo

Vitória, ES  
2021

*Dedico este trabalho à minha família, amigos e professores que me deram todo o suporte e conhecimento para chegar até aqui.*

# Agradecimentos

Agradeço em primeiro lugar, a Deus, que permitiu que eu tivesse saúde e determinação, por me permitir ultrapassar todos os obstáculos encontrados, e que fez com que meus objetivos fossem alcançados durante todos os meus anos de estudos.

Aos meus familiares, que me incentivaram nos momentos difíceis, por todo o apoio e pela ajuda, que muito contribuíram para a realização deste trabalho, sendo uma base para que eu pudesse chegar até aqui.

Aos amigos da vida, que sempre estiveram ao meu lado, pela amizade incondicional e pelo apoio demonstrado, que tornaram o caminho percorrido mais fácil.

Aos meus colegas de curso, com quem convivi intensamente durante os últimos anos, pelo companheirismo e pela troca de experiências que me permitiram crescer não só como pessoa, mas também como formando.

À UFES, essencial no meu processo de formação profissional, pela dedicação, e por tudo o que aprendi ao longo dos anos do curso.

Ao Professor Flávio Miguel Varejão agradeço imensamente por ter sido meu orientador e ter desempenhado tal função com tamanha dedicação, pelo suporte, paciência e todo conhecimento compartilhado.

Agradeço ao Prof. Thomas Walter Rauber e ao Prof. Thiago Oliveira dos Santos pela disponibilidade e por aceitarem fazer parte da banca, contribuindo com o enriquecimento do trabalho desenvolvido.

A todos aqueles que contribuíram, de alguma forma, para a realização deste trabalho.

*“A persistência é o caminho do êxito.”  
(Charles Chaplin)*

# Resumo

No mercado de apostas esportivas existem diversas variáveis que são usadas pelas casas de apostas para estipularem uma probabilidade. Devido a este fato alinhado com a recente entrada dos esportes eletrônicos neste meio, muita das vezes a probabilidade dada pode não condizer com a real chance, levantando a questão se pode uma IA estipular a real chance de determinada aposta.

Uma das grandes vantagens dos esportes eletrônicos neste sentido, é o fato de que os dados estatísticos são de bem mais fácil acesso, no caso do League of Legends, outro ponto a se considerar seria que, existem variáveis que influenciam diretamente no favoritismo de uma equipe

Neste contexto, este Trabalho de Conclusão de Curso aborda o desenvolvimento de um Sistema de Predição de Resultados de Apostas Esportivas, onde o projeto visa a utilização de um modelo de classificação para identificar em uma partida qual equipe sairá vencedora.

No intuito de construir um modelo com o maior desempenho possível, são realizados experimentos e otimizações com diversas técnicas de *machine learning* com a finalidade de comparar seus resultados, e obter o modelo final ideal.

Comparamos nosso resultado final com os modelos anteriores, a fim de compreender melhor o desempenho preditivo. Descobrimos que nosso modelo final apresenta acurácia satisfatoriamente melhor do que sua versão não otimizada, assim como suas versões utilizando somente dados em tempo real, e utilizando somente dados pré-jogo.

Desta forma, este trabalho apresenta uma aplicação web que auxilia apostadores a visualizarem de forma amigável as previsões feitas pelo modelo em tempo real.

**Palavras-chaves:** *machine learning*, classificação, apostas esportivas, sistema, web.



# Lista de ilustrações

Figura 1 – Arquitetura geral . . . . .	28
Figura 2 – Arquitetura Geral front-end . . . . .	29
Figura 3 – Arquitetura Detalhada front-end . . . . .	30
Figura 4 – Arquitetura back-end . . . . .	31
Figura 5 – Arquitetura Modelo . . . . .	32
Figura 6 – Arquitetura Modelo Validação . . . . .	33
Figura 7 – Interface Horários . . . . .	34
Figura 8 – Interface Pré-Jogo . . . . .	34
Figura 9 – Componente Pré-Jogo - Split . . . . .	35
Figura 10 – Componente Estatísticas Time Azul . . . . .	36
Figura 11 – Componente Estatísticas Time Vermelho . . . . .	36
Figura 12 – Interface Ao Vivo . . . . .	37
Figura 13 – Componente Ao Vivo Estatísticas . . . . .	37
Figura 14 – Componente Predição 10 Minutos . . . . .	38
Figura 15 – Componente Predição 15 Minutos . . . . .	38
Figura 16 – Componente Tempo Real Time Azul . . . . .	39
Figura 17 – Componente Tempo Real Time Vermelho . . . . .	39
Figura 18 – Resultados kNN . . . . .	47
Figura 19 – Resultados Variantes kNN . . . . .	48
Figura 20 – Resultados Naive Bayes . . . . .	49
Figura 21 – Resultados Variantes Naive Bayes . . . . .	50
Figura 22 – Resultados Logistic Regression . . . . .	51
Figura 23 – Resultados Variantes Logistic Regression . . . . .	52
Figura 24 – Resultados Random Forest . . . . .	53
Figura 25 – Resultados Variantes Random Forest . . . . .	54
Figura 26 – Resultados Aos 10 Minutos . . . . .	55
Figura 27 – Resultados Aos 15 Minutos . . . . .	55
Figura 28 – Resultados Otimizados Naive Bayes . . . . .	57
Figura 29 – Resultados Otimizados Logistic Regression . . . . .	58
Figura 30 – Resultados Otimizados Random Forest . . . . .	59
Figura 31 – Resultados Otimizados Aos 10 Minutos . . . . .	60
Figura 32 – Resultados Otimizados Aos 15 Minutos . . . . .	60
Figura 33 – Comparações Gerais . . . . .	61

# Lista de tabelas

Tabela 1 – RF01 . . . . .	25
Tabela 2 – RF02 . . . . .	25
Tabela 3 – RF03 . . . . .	26
Tabela 4 – RF04 . . . . .	26
Tabela 5 – RNF01 . . . . .	27
Tabela 6 – RNF01 . . . . .	27
Tabela 7 – Melhores Parâmetros - Naive Bayes . . . . .	57
Tabela 8 – Melhores Parâmetros - Logistic Regression . . . . .	58
Tabela 9 – Melhores Parâmetros - Random Forest . . . . .	59

# Lista de abreviaturas e siglas

LoL	<i>League of Legends</i>
IA	Inteligência Artificial
RF	Requisito Funcional
RNF	Requisito Não Funcional
API	<i>Application Programming Interface</i>
TP	<i>True Positive</i>
TN	<i>False Positive</i>
FP	<i>True Negative</i>
FN	<i>False Negative</i>
kNN	<i>k-Nearest-Neighbors</i>

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>13</b>
<b>1.1</b>	<b>Motivação e Justificativa</b>	<b>13</b>
<b>1.2</b>	<b>Objetivos</b>	<b>14</b>
<b>1.3</b>	<b>Organização da Monografia</b>	<b>14</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA E TECNOLOGIAS UTILIZADAS</b>	<b>16</b>
<b>2.1</b>	<b>Mercado de apostas</b>	<b>16</b>
2.1.1	Visão Geral	16
2.1.2	Como Funciona	17
2.1.3	Estratégia Lucrativa	17
<b>2.2</b>	<b>League Of Legends e E-sports</b>	<b>18</b>
2.2.1	O Jogo	18
2.2.2	E-sports	18
2.2.3	Mercado de Apostas	19
<b>2.3</b>	<b>Machine learning e Predição</b>	<b>19</b>
2.3.1	Modelos Clássicos	20
2.3.2	Machine learning	20
2.3.3	Sistemas de Predição	21
<b>2.4</b>	<b>Tecnologias e Ferramentas</b>	<b>22</b>
2.4.1	Aplicação Web	22
2.4.2	Modelo Preditivo	23
<b>3</b>	<b>SISTEMA DE PREDIÇÃO DE RESULTADOS DE APOSTAS ES- PORTIVAS</b>	<b>24</b>
<b>3.1</b>	<b>Requisitos</b>	<b>24</b>
3.1.1	Requisitos Funcionais	25
3.1.2	Requisitos Não Funcionais	26
<b>3.2</b>	<b>Arquitetura</b>	<b>27</b>
3.2.1	Camada Front-end	28
3.2.2	Camada Back-end	30
3.2.3	Camada Modelo	31
<b>3.3</b>	<b>Telas do Sistema e Suas Funcionalidades</b>	<b>33</b>
3.3.1	Tela Horários	34
3.3.2	Tela Pré-Jogo	34
3.3.3	Tela Ao Vivo	37

<b>4</b>	<b>AVALIAÇÃO DA PROPOSTA</b>	<b>40</b>
<b>4.1</b>	<b>Conjunto de Dados</b>	<b>40</b>
4.1.1	Descrição	40
4.1.2	Preparação	41
4.1.3	Variáveis	42
<b>4.2</b>	<b>Experimentos</b>	<b>44</b>
4.2.1	Repeated k-Fold Cross-Validation	44
4.2.2	Métricas	45
4.2.3	Modelos de Classificação	45
4.2.4	k-Nearest-Neighbors	46
4.2.5	Naive Bayes	48
4.2.6	Logistic Regression	50
4.2.7	Random Forest	52
4.2.8	Escolha dos Modelos	54
<b>4.3</b>	<b>Otimização</b>	<b>56</b>
4.3.1	Grid Search	56
4.3.2	Naive Bayes	56
4.3.3	Logistic Regression	57
4.3.4	Random Forest	58
4.3.5	Modelo Final	60
4.3.6	Comparações Gerais	61
<b>4.4</b>	<b>Avaliação de Lucro nas Apostas</b>	<b>62</b>
<b>5</b>	<b>CONCLUSÃO</b>	<b>63</b>
<b>5.1</b>	<b>Considerações Finais</b>	<b>63</b>
<b>5.2</b>	<b>Trabalhos Futuros</b>	<b>64</b>
5.2.1	Melhores Conjuntos de Dados	64
5.2.2	Modelos Preditivos Para Outros Mercados	64
5.2.3	Validação Lucrativa das Apostas	65
5.2.4	Sistema de Automação de Apostas	65
	<b>REFERÊNCIAS</b>	<b>66</b>

# 1 Introdução

Os sistemas de previsão estão revolucionando o mundo da tecnologia, permitindo resolver problemas que anteriormente não eram capazes de serem solucionados. Nos esportes, esses sistemas podem ser utilizados em diversas áreas, para os treinadores analisarem o desempenho do time, definir seu plano de jogo, contratar novos jogadores, e também para o mercado de apostas.

As apostas esportivas partem de uma premissa muito simples: você aposta que um determinado evento irá ocorrer em uma partida (vitória de uma equipe, empate, derrota) e, caso ocorra, você recebe o que apostou mais um valor extra, caso tal não ocorra, você perde. Diante de cada evento, existe uma cotação para o mesmo, que pode variar de acordo com o risco envolvido, se um evento é altamente improvável de acontecer, ele terá uma rentabilidade maior caso o apostador vença.

Nos últimos anos esse mercado vem crescendo como nunca antes visto, e hoje é possível apostar nos mais diversos esportes, como por exemplo esportes mais tradicionais (Futebol, Basquete, Futebol Americano) e também esportes eletrônicos (*Counter Strike*, *DOTA*, *League Of Legends*).

Os esportes eletrônicos, também chamados de *e-sports*, são uma nova modalidade esportiva que surgiu há poucos anos. Onde existem competições, em que equipes que possuem uma estrutura profissional competem entre si.

Diferente de esportes mais tradicionais, os *e-sports* ainda não estão totalmente difundidos no meio de apostas esportivas, apesar disso, hoje movimentam centenas de milhares de dólares somente nesta esfera esportiva. Sendo o *League Of Legends* atualmente o *e-sport* principal, com o maior público e os torneios mais bem estruturados.

## 1.1 Motivação e Justificativa

No ramo de apostas esportivas existem muitas variáveis que são usadas pelas casas de apostas para estipularem uma probabilidade. Devido a este fato alinhado com a recente entrada dos esportes eletrônicos neste meio, muita das vezes a probabilidade dada pode não condizer com a real chance, levantando a questão se pode uma IA estipular a real chance de determinada aposta. Podendo destacar alguns fatores que indicam a possibilidade de sucesso na aplicação da ideia:

- Diferente dos esportes mais tradicionais, por ser algo mais recente, o *League Of Legends* ainda não está totalmente difundido nesse meio de apostas esportivas. Devido a isto,

alinhado com a necessidade repentina das casas de apostas disponibilizarem esse tipo de mercado, se abrem possibilidades para as casas não estipularem corretamente as probabilidades.

- Outro ponto a se considerar seria o fato de que, em jogos como o *League Of Legends*, existem variáveis que influenciam diretamente no favoritismo de uma equipe e juntamente com as dificuldades significativas para as casas de apostas em estipular as probabilidades para as partidas, se abre um espaço interessante para gerar lucratividade.
- Uma das grandes dificuldades na utilização de *machine learning* para predição é o acesso aos dados correlatos ao problema a ser resolvido, como o *League Of Legends* é um esporte eletrônico, os dados estatísticos são de bem mais fácil acesso. Sendo disponibilizado pela própria desenvolvedora do jogo, a *Riot Games*.

## 1.2 Objetivos

O objetivo geral deste trabalho será desenvolver uma aplicação web que utilize um modelo de análise preditiva criado a partir de técnicas de *machine learning* para identificar em uma partida qual equipe sairá vencedora, e com isso, usar esta informação para tomar uma decisão na aposta.

São objetivos específicos deste trabalho:

- Desenvolver um modelo de classificação que seja capaz de prever o resultado da partida.
- Construir uma aplicação web que possibilite a integração com a IA criada e apresente os resultados de forma amigável.
- Orientar o investidor a realizar as apostas utilizando um método que seja capaz de gerar lucro.

## 1.3 Organização da Monografia

Esta monografia está dividida em 5 capítulos. O Capítulo atual faz uma introdução ao problema, apresentando as motivações para o desenvolvimento, quais são os objetivos e o método no qual o trabalho foi desenvolvido.

O capítulo 2 apresenta os aspectos relativos ao conteúdo teórico relevante para o trabalho, abordando para isto uma visão geral sobre mercado de apostas, como ele funciona, e como construir uma estratégia que seja capaz de gerar lucro.

O capítulo 3 apresenta os requisitos funcionais e não funcionais do sistema, como a arquitetura do sistema foi projetada, assim como o funcionamento dos seus módulos, e por fim serão expostas as telas do sistema e suas responsabilidades.

O capítulo 4 apresenta a avaliação do modelo de predição, detalhando sobre o conjunto de dados que foi utilizado no projeto, explicando o método de teste utilizado para avaliar, assim como os diferentes modelos de classificação que foram treinados, e por fim é apresentado o método utilizado para realizar as otimizações nos modelos, exibindo os respectivos resultados.

E finalmente no capítulo 5, é traçada uma conclusão sobre o estudo, apresentando as dificuldades e limitações enfrentadas e feita a sugestões para trabalhos futuros.



## 2 Fundamentação Teórica e Tecnologias Utilizadas

O Capítulo 2 apresenta a base de conceitos e tecnologias que fundamentaram e tornaram possíveis o desenvolvimento do projeto, e está organizado da seguinte forma: a Seção 2.1 apresenta como funciona o mercado, o fenômeno social, estratégias lucrativas e métodos de aposta; a Seção 2.2 apresenta as regras do jogo, eventos que ocorrem durante a partida, os resultados possíveis e um paralelo com o mercado de apostas em geral; já a Seção 2.3 apresenta uma visão geral do que é *machine learning*, assim como introduz sobre modelos clássicos de predição, traçando um paralelo com sistemas de predição para problemas como o de prever apostas; a Seção 2.4 irá definir quais tecnologias e ferramentas serão utilizadas, apresentando como será implementado o sistema web e o modelo de predição.

### 2.1 Mercado de apostas

O mercado online de apostas esportivas está em plena ascensão e vem ganhando espaço e popularidade no cenário mundial nos últimos anos. Nesta seção iremos apresentar uma visão geral sobre as apostas esportivas, assim como uma explicação sobre como funciona, e por fim será exposta uma construção de uma estratégia que seja capaz de gerar lucro.

#### 2.1.1 Visão Geral

As apostas sempre foram vistas como um jogo de azar pela população em geral, porém, nos últimos anos essa visão tem mudado e é possível notar um grande crescimento do mercado de apostas esportivas no Brasil. Uma área, que antes era vista apenas como uma opção de lazer, virou a profissão de muita gente, que hoje vive apenas dos ganhos gerados pelas apostas.

Apesar de ainda não ser regulamentada no Brasil, as apostas esportivas estão oficialmente legalizadas no país desde o mês de dezembro de 2018, com isso as casas de apostas deixam de ser taxadas como atividade criminosa. Devido a este fato, tem alcançado um nível no Brasil que já podia ser visto em outros locais há alguns anos, como a Europa, por exemplo, onde as casas de apostas patrocinam grandes competições e clubes.

Com o futebol sendo 70% do foco principal de apostadores, outros esportes como basquete, corrida de cavalos, futebol americano e também os *e-sports* crescem em um grande ritmo e prometem ser ainda mais fortes com aumento da visibilidade e receita na casa dos milhões.

### 2.1.2 Como Funciona

O mercado esportivo nada mais é do que uma negociação de probabilidades durante determinado evento esportivo, ou seja, dentro de cada partida (basquete, vôlei, *League Of Legends*) existem pessoas comprando e vendendo as possibilidades envolvidas nesses eventos.

As apostas possuem uma dinâmica bem parecida de como ocorre na bolsa de valores, para que se ocorra uma negociação, é preciso haver um comprador e um vendedor, isto é, para você comprar uma ação precisa ter alguém querendo vendê-la no mesmo preço. O trabalho do *trader*, na bolsa de valores, é traçar a melhor estratégia entre o valor de compra de uma ação e o valor de venda, assim como o apostador deve traçar uma estratégia onde haverá uma relação lucrativa entre a compra da aposta e a venda.

Diante de cada evento, existe uma cotação para o mesmo, que pode variar de acordo com o risco envolvido. O risco tem relação com a probabilidade, se um evento é altamente improvável de acontecer, ele sempre vai ter uma rentabilidade maior caso o apostador vença.

### 2.1.3 Estratégia Lucrativa

Como discorreremos a relação entre a bolsa de valores e o mercado esportivo na subseção 2.1.2, as nossas estratégias serão definidas a partir de probabilidades e estatísticas. Ao contrário de apostadores em geral que realiza suas apostas em eventos que ele acha que vai acontecer, o nosso trabalho é encontrar assimetrias em relação as probabilidades oferecidas pelas casas de apostas e as probabilidades reais para um evento.

Dessa forma, a aposta deixa de ser meramente uma especulação e passa a uma busca por valor nas apostas, ou seja, encontrar probabilidades oferecidas pelas casas de apostas que sejam menores que as probabilidades reais para um evento, sendo que, quanto maior a probabilidade real, maior será o valor encontrado e com isso maior será o lucro.

Por exemplo, você decide apostar em qual time será vencedor em uma partida de *League Of Legends*, e ao analisar, percebe que a casa de apostas está pagando como se o time tivesse apenas 50% de chance de vencer, mas você sabe que ele tem uma chance muito melhor de vencer com 75%. Lembrando que as apostas esportivas lhe pagarão mais dinheiro quando algo for menos provável de acontecer. Isso significa que, como eles acham que a aposta tem apenas 50% de chance, eles vão oferecer um pagamento muito melhor do que se a casa estipulasse corretamente a probabilidade.

Tendo isso em mente, caso sejam sempre feitas apostas com valor encontrado, por mais que não acerte todos os resultados e até mesmo tenha prejuízo em uma pequena quantidade inicial de partidas, ao longo prazo terá lucro, já que como suas apostas tem mais chance de acontecer do que a estipulada pela casa, o lucro potencial em cima dessas apostas é maior do que deveria.

## 2.2 League Of Legends e E-sports

Esportes Eletrônicos ou *e-sports* são uma nova modalidade de esportes surgida há poucos anos e que vêm dominando o mercado de games, tendo como expoente o *League Of Legends*. Nesta seção será apresentado as regras do jogo, eventos que ocorrem durante a partida, assim como uma visão geral sobre o *e-sports* e o atual cenário em que se encontra, e por fim será introduzido um paralelo com o mercado de apostas em geral.

### 2.2.1 O Jogo

*League Of Legends* é um jogo do gênero *Multiplayer Online Battle Arena (MOBA)*, onde duas equipes, compostas por cinco jogadores cada, competem entre si para ser a primeira equipe a destruir o *Nexus* da equipe adversária. Em seu caminho estão não apenas os 5 jogadores do time adversário, mas também várias unidades controladas por computador, chamadas de *Minions* e as Torres.

Os jogadores controlam unidades chamadas de Campeões, e eles devem destruir as Torres e Inibidores inimigos para chegar ao *Nexus*. Eliminar as "peças" adversárias concede ao jogador recursos no jogo que tornam os mais fortes. Uma vez que um *Nexus* é destruído, a equipe é declarada a vencedora da partida.

### 2.2.2 E-sports

Jogos eletrônicos são uma forma de diversão que inerentemente envolve competição em níveis mais ou menos definidos. Podemos disputar partidas contra amigos ou desconhecidos, online ou presencialmente, até mesmo quando estamos sozinhos, estamos competindo contra a máquina, contra o tempo, e até contra nós mesmos.

Antigamente o acesso ao entretenimento e ao esporte era algo completamente passivo, sujeito às programações dos canais de TV. No entanto com a evolução da tecnologia, o que eram competições simples contra a máquina ou partidas contra amigos nos últimos anos passou a se tornar uma área que movimenta bilhões de dólares em que estão envolvidos times, jogadores, empresas dos jogos, serviços de *live streaming* e empresas que apoiam esse mercado a crescer.

Os campeonatos são extremamente estruturados e distribuindo premiações milionárias. Segundo o site Esports Earnings, voltado para pesquisa de valores dos campeonatos da modalidade, o DOTA 2 é o principal jogo do mundo em termos de premiação, tendo desembolsado mais de 225 milhões de dólares em cerca de 1400 torneios.

O Counter-Strike: Global Offensive (CS:GO) também possui destaque na distribuição de premiação, com mais de 99 milhões de dólares em cerca de 5000 campeonatos. A terceira posição é ocupada pelo Fortnite, com cerca de 90 milhões de dólares em apenas 564 tor-

neios, seguido pelo LoL, que distribuiu em torno de 76 milhões de dólares em mais de 2400 competições.

Os campeonatos de *e-sports* são assistidos por milhões de pessoas em todo mundo. Em 2019, segundo a Newzoo, a audiência dos jogos eletrônicos foi de 453,8 milhões de pessoas, em um crescimento de 12,3% em relação a 2018. Já para 2021, a previsão é de que o público global de *live streaming* atingirá 729 milhões de pessoas em 2021

A indústria dos *e-sports* também está diretamente relacionada ao desenvolvimento de uma outra indústria: a dos serviços de *live streaming*, com plataformas como Twitch, Facebook, Nimo TV e YouTube sendo usadas para a transmissão de partidas dos campeonatos profissionais em tempo real. Esse ecossistema, porém, também gira fora dos campeonatos, com os *streamers* constituindo uma outra classe de carreira. Não necessariamente atletas profissionais, eles movimentam a internet com transmissões, análises e conteúdos variados.

Um dos expoentes dos *e-sports* é o *League Of Legends*, que é hoje o principal nome nesse meio, seja pela sua estrutura organizacional e o dinheiro movimentado, que possui ligas ao redor de todo o mundo, como é no caso do CBLOL (Campeonato Brasileiro de *League Of Legends*).

### 2.2.3 Mercado de Apostas

O universo dos *e-sports* se desenvolve constantemente, como citado na subseção 2.2.2, junto dele as apostas esportivas também se mostram em ascensão entre os populares 2.1.1. Devido a isto muitas casas de apostas não ignoraram tais fatos e, nos últimos anos, permitiram que os apostadores apostassem em várias modalidades de *e-sports*

Pela perspectiva das casas de apostas, o cenário de *e-sports* ainda está sendo estudado, e uma das vantagens para os apostadores são os erros na estipulação das probabilidades, pois isso facilita perceber esses erros e encontrar valor nessas apostas.

Outro ponto a se considerar pelos apostadores seria o fato de que, em certos jogos, como o *League Of Legends*, existem variáveis que influenciam diretamente no favoritismo de uma equipe e juntamente com as dificuldades significativas para as casas de apostas em definir cotações para partidas em andamento, se abre um espaço interessante para gerar lucratividade.

## 2.3 Machine learning e Predição

*Machine learning* é um método de análise de dados que automatiza a construção de modelos analíticos que são capazes de realizar predições. Nesta seção será apresentado uma breve introdução sobre modelos clássicos de predição, assim como uma visão geral sobre *machine learning*, e por fim será apresentado o conceito de sistemas de predição, traçando

um paralelo com o problema de prever apostas.

### 2.3.1 Modelos Clássicos

A tentativa de prever resultados esportivos é algo que tem sido tema de pesquisa desde meados do século 20, com as primeiras abordagens utilizando modelos estatísticos como de Moroney (1956) e Reep (1971) que usaram tanto a distribuição de Poisson quanto a distribuição binomial negativa para modelar com base em resultados anteriores de times a quantidade de gols que viriam a ser marcados em uma partida de futebol.

O primeiro grande avanço veio de Maher (1982), que usou Poisson para modelar as capacidades defensivas e de ataque das equipes que disputariam a partida, e usou isso para prever o número médio de gols de cada equipe. Em seguida, Dixon e Coles (1997) foi o primeiro a criar um modelo capaz de gerar probabilidades para resultados de partidas e pontuações, novamente seguindo uma distribuição de Poisson. Modelo esse que ainda hoje é visto como um modelo de sucesso, e é utilizado como uma referência em relação aos modelos de *machine learning*.

### 2.3.2 Machine learning

Podemos definir de forma simples *machine learning* como o processo pelo qual os computadores desenvolvem o reconhecimento de padrões com a capacidade de aprender continuamente ou fazer previsões baseadas em dados, e então, fazer ajustes sem serem especificamente programados para essa tarefa.

O aprendizado de máquina pode usar uma variedade de algoritmos que de forma autônoma aprende com os dados e gera uma saída para melhorar, descrever dados ou prever resultados, e a medida que os algoritmos ingerem dados de treinamento, é possível produzir modelos mais precisos com base nesses dados.

Dependendo da natureza do problema em questão, existem algumas abordagens de *machine learning*, com base por exemplo no tipo de dado e a quantidade disponível. Hoje apesar de não serem os únicos, dois métodos são os mais adotados, que são o aprendizado supervisionado e o aprendizado não-supervisionado.

A aprendizagem não-supervisionada geralmente é utilizada contra dados que não possuem rótulos definidos, ou seja, a resposta certa não é informada. Portanto o algoritmo deve descobrir o que está sendo mostrado. O objetivo é explorar os dados e encontrar alguma informação dentro deles. A aprendizagem não-supervisionada funciona bem por exemplo, em identificar usuários com atributos parecidos e que podem, então, ser tratados de modo similar, ou também pode encontrar os principais atributos que separam segmentos distintos de usuários.

É considerado uma aprendizagem supervisionada quando é dado um conjunto de dados rotulados que já sabemos qual é a saída correta, tendo a ideia de que existe uma relação entre a entrada e a saída. Esses problemas de aprendizagem supervisionados são classificados em problemas de classificação e regressão.

No caso de prever o resultado de uma partida de *League Of Legends*, estamos tentando prever os resultados em uma saída discreta (Vitória Time Azul / Vitória Time Vermelho), ou seja, é um problema de classificação. Neste caso procura-se estimar um “classificador” que gere como saída a classificação qualitativa de um dado não observado com base nos dados de entrada.

A regressão por outro lado, de forma similar a classificação, utiliza dados de entrada já observados para prever uma resposta. A grande diferença é que, neste caso, procura-se estimar um valor numérico e não uma classificação de uma observação.

O uso do *machine learning* nas mais diversas aplicações só tende a crescer nos próximos anos, porém, muitos recursos tecnológicos que temos hoje só funcionam ou são viáveis por conta da inteligência artificial. Como por exemplo:

- Combate a fraudes em sistemas de pagamento, onde diversas tentativas de fraudes com cartões de crédito e outros meios de pagamentos são geradas a cada segundo no mundo todo, e com *machine learning* os sistemas de combate a fraudes conseguem impedir a maior parte dessas ações;
- Recomendação de conteúdo, onde as plataformas de *streaming* de vídeo usam os modelos de *machine learning* para analisar o histórico de conteúdo consumido pelo usuário, e a partir de escolhas feitas pelo mesmo passa a dar a ele recomendações que possuem correlação com os seus gostos.
- Reconhecimento de doenças, onde alguns hospitais modernos, os quais são equipados com aparelhos que reúnem e compartilham grandes quantidades de dados, utilizam esses para análises clínicas utilizando inteligência artificial. Um exemplo de destaque que apresenta capacidade relativamente maior do que médicos humanos é na detecção de câncer de pele.

### 2.3.3 Sistemas de Predição

Com o avanço da tecnologia, alinhado com o surgimento de técnicas de *machine learning* e o aumento de dados disponíveis, os sistemas de predição voltaram a ser uma área em destaque.

Tempos atrás os conceitos de *machine learning* e análise preditiva eram vistos como totalmente diferentes e não relacionados, porém com a demanda crescente de análises de dados nos dias de hoje, as técnicas de *machine learning* caminharam para resolver as

análises preditivas. Hoje, os sistemas preditivos usam quase em sua totalidade *machine learning* devido à sua capacidade de reconhecer padrões e processar com precisão grandes quantidades de dados.

Com o avanço da Inteligência Artificial, os sistemas de previsão agora são capazes de resolver problemas que antes não era possível. Nos esportes, esses sistemas podem ser utilizados em diversas áreas, as equipes realizarem *scouting*, para os treinadores analisarem o desempenho do time, definir seu plano de jogo, e também para estipular o resultado de uma partida.

As apostas esportivas se encaixam perfeitamente em um desses problemas para sistemas de predição, com uma grande quantidade de dados disponíveis e um objetivo simples de escolher o vencedor. No entanto, existe algo específico neste tipo de problema, que é o fato de que não estamos tentando somente prever o resultado do jogo, mas também se tal previsão será uma aposta lucrativa.

É importante levar em consideração que nem todas as apostas oferecem a mesma recompensa. Uma aposta de R\$1,00 com odds de 2 pode gerar R\$1,00 de lucro, enquanto odds de 1,1 geraria um lucro bem menor de R\$0,01, sendo que ambas as apostas possuem a mesma perda de R\$1,00 se não forem bem-sucedidas. Portanto, o sistema de predição a partir da probabilidade de acerto deve se preocupar em encontrar apostas em que irão pagar mais do que realmente elas deveriam pagar conforme foi discorrido na seção 2.1.3.

## 2.4 Tecnologias e Ferramentas

Nesta seção será definido quais tecnologias e ferramentas que serão utilizadas no trabalho, apresentando como será implementado o sistema web e o modelo preditivo.

### 2.4.1 Aplicação Web

A aplicação web será implementada através do ambiente Visual Studio Code, que é um editor de código-fonte desenvolvido pela Microsoft, juntamente com *plugins* disponíveis, que podem ser integrados ao editor, com isso teremos um ambiente com todas as ferramentas necessárias para o desenvolvimento.

A linguagem de programação escolhida para desenvolver a aplicação será o JavaScript, que com a ajuda da biblioteca React, o desenvolvimento se tornará bem mais fácil e eficiente. A principal característica do React é permitir a criação de componentes que podem ser reutilizados em diversas páginas. Na prática, são blocos de códigos reutilizáveis que agregam funcionalidades e que retornam ao código HTML após a renderização pelo navegador, dessa forma, ganha-se em produtividade e facilidade para a manutenção.

Existem outras ferramentas, *frameworks* e *plugins* que auxiliam no desenvolvimento

com o React. Como é o caso do framework Material-Ui que fornece componentes estilizados sob o Material Design, e o Redux que é uma biblioteca desenvolvida em JavaScript e utilizada para garantir que o estado de um elemento utilizado em diferentes views seja consistente, permitindo que o estado do componente React seja alterado em apenas de forma global. e por último o Axios que é um cliente HTTP baseado em promises para fazer requisições, ele será utilizado para capturar os dados em tempo real que serão utilizados pelos modelos preditivos, e serão oferecidos pela empresa responsável pelo desenvolvimento do jogo *League Of Legends*.

### 2.4.2 Modelo Preditivo

O modelo preditivo será implementado através da plataforma Jupyter, ferramenta computacional gratuita e de fácil instalação, que permite gerir distribuições de Python, ambientes de trabalho e módulos, visando simplificar o gerenciamento e implantação de pacotes.

Algumas bibliotecas irão nos auxiliar no desenvolvimento, como o Pandas que é uma biblioteca criada para a linguagem Python para manipulação e análise de dados, oferecendo estruturas e operações para manipular tabelas e séries, junto dela será utilizada o NumPy , uma poderosa biblioteca que é usada principalmente para realizar cálculos em Arrays e Matrizes. Para visualizar os dados disponíveis Seaborn sera a biblioteca, baseada em matplotlib, ela fornece uma interface de alto nível para desenhar gráficos estatísticos.

Os algoritmos de *machine learning* serão disponibilizados pela scikit-learn, uma biblioteca de aprendizado de máquina de código aberto para a linguagem de programação Python. Ela inclui vários algoritmos de classificação, regressão e agrupamento incluindo Random Forest, Dummy (ZeroR), K-Nearest Neighbors, Gaussian Naive Bayes, e é projetada para interagir com as bibliotecas Python numéricas e como a NumPy.



## 3 Sistema de Predição de Resultados de Apostas Esportivas

Visando a utilização do modelo preditivo no contexto de uso por meio de um sistema web, esta seção se concentra em documentar o processo inicial para o desenvolvimento. Esta primeira etapa busca investigar o problema com o objetivo de ter uma visão mais ampla sobre o sistema. A análise bem detalhada unida a um projeto bem elaborado facilita as etapas seguintes do desenvolvimento de *software*, assim como uma interpretação equivocada pode levar a problemas durante a fase de implementação.

Neste capítulo, faremos uma breve apresentação dos requisitos funcionais e não funcionais do sistema, também será exibida a arquitetura e detalhados o funcionamento dos seus módulos, e por fim serão expostas as telas do sistema juntamente com as suas respectivas responsabilidades.

### 3.1 Requisitos

Quando planejamos o desenvolvimento de um *software*, o primeiro passo é analisar e descrever os detalhes do que ele pretende fazer, considerando restrições possíveis, limitações de recursos de acordo com o contexto de desenvolvimento. Esse processo de analisar, pesquisar, verificar e descrever é chamado de requisito de *software* (Sommerville, 2010).

De maneira mais geral, requisito é um aspecto que o sistema proposto deve fazer ou uma restrição no desenvolvimento do sistema, logo os requisitos são necessários para cumprir as expectativas, portanto a sua especificação é uma tarefa crucial no desenvolvimento do sistema, já que requisitos pouco claros ou não definidos podem causar retrabalhos, atrasos, ou até uma possível inviabilização do projeto.

Geralmente, os requisitos podem ser divididos em duas categorias: requisitos funcionais e requisitos não funcionais. O primeiro se refere ao comportamento do sistema, seus requisitos para o funcionamento de cada tarefa, ou seja, eles devem documentar como o sistema deve lidar e se comportar, o segundo embora não possua uma definição tão clara, podemos dizer que definem como o sistema fará, não estão relacionados diretamente às funcionalidades, são responsáveis por tratar das restrições de recursos computacionais, por exemplo.

### 3.1.1 Requisitos Funcionais

Os requisitos funcionais são de extrema importância no desenvolvimento de aplicativos, pois sem eles não há funcionalidades nos sistemas, como dito antes, eles são responsáveis por descrever os serviços e recursos que o *software* deve apresentar.

Para isso, é preciso que esteja bem explicado e sem contradições, em outras palavras, sua modelagem deve ser feita de forma clara e objetiva, além disso sua codificação deve ser totalmente aplicável.

Tendo esta definição em mente, seguem abaixo os requisitos funcionais do *software*:

Tabela 1 – RF01

Nome do requisito	Exibir próximas partidas
Identificador	RF01
Prioridade	Alta
Complexidade	Média
Descrição do requisito	O sistema deverá permitir ao usuário visualizar as próximas partidas que ocorrerão separadas pelos seus respectivos dias.

Tabela 2 – RF02

Nome do requisito	Exibir estatísticas dos times
Identificador	RF02
Prioridade	Alta
Complexidade	Complexa
Descrição do requisito	O sistema deverá permitir ao usuário visualizar as estatísticas da temporada/ano dos times que irão disputar alguma partida no dia.

Tabela 3 – RF03

Nome do requisito	Exibir estatísticas ao vivo
Identificador	RF03
Prioridade	Alta
Complexidade	Complexa
Descrição do requisito	O sistema deverá permitir ao usuário visualizar as estatísticas ao vivo dos times que estão disputando alguma partida no momento.

Tabela 4 – RF04

Nome do requisito	Exibir predição do vencedor
Identificador	RF04
Prioridade	Alta
Complexidade	Complexa
Descrição do requisito	O sistema deverá permitir ao usuário visualizar a predição feita sobre qual time será o vencedor de determinada partida.

### 3.1.2 Requisitos Não Funcionais

Os requisitos não funcionais são tratados geralmente como premissas e restrições técnicas de um projeto, são praticamente todas as necessidades que não podem ser atendidas através de funcionalidades.

Os requisitos não funcionais descrevem o sistema em termos de desempenho, confiança, usabilidade e tecnologia. Geralmente, são tidos como pontos críticos na aplicação, pois a definição inadequada desses requisitos impossibilita o uso correto da aplicação, mesmo se os requisitos funcionais estiverem corretos.

Tendo esta definição em mente, seguem abaixo os requisitos não funcionais do *software*:

Tabela 5 – RNF01

Nome do requisito	Predição de resultado
Identificador	RNF01
Prioridade	Alta
Complexidade	Complexa
Tipo do requisito	Confiabilidade
Descrição do requisito	As predições das partidas precisam ter entre 75% a 85% de acertos para cada usuário.

Tabela 6 – RNF01

Nome do requisito	Predição de resultado
Identificador	RNF01
Prioridade	Alta
Complexidade	Complexa
Tipo do requisito	Desempenho
Descrição do requisito	As predições das partidas aos 10 minutos e aos 15 minutos devem ser exibidas aos usuários em menos de 10 segundos.

## 3.2 Arquitetura

A arquitetura de *software* de um sistema abrange a forma como suas partes são organizadas, incluindo questões como o comportamento dessa estrutura e quais componentes são responsáveis por realizar um conjunto específico de funções. Resumidamente, é um modelo repetível sob o qual um sistema pode ser desenvolvido,

A escolha de uma arquitetura influencia aspectos como a performance, qualidade, facilidade de manutenção e escalabilidade, assim, essa decisão tem grande impacto no sucesso do projeto, principalmente a longo prazo. (Bosch, 2000)

Existem diversos princípios e padrões que são utilizados nos sistemas e normalmente os projetos desenvolvidos não se limitam a um único estilo ou arquitetura. Em vez disso, são uma combinação de padrões que, juntos, formam o sistema completo.

A arquitetura em camadas é o padrão de arquitetura de *software* mais utilizado. E foi o selecionado para ser utilizado nesse sistema. Aqui, os componentes serão organizados em camadas horizontais e interconectados, mas independentes entre si. Os módulos do *software*

são organizados separados pelas suas funcionalidades, que podem ser desconstruídas em diferentes serviços.

De forma simplista a figura 1 mostra como ficou configurada as camadas do sistema, onde cada módulo ficou focado em uma tarefa macro. Logo pode-se dividir-se o desenvolvimento em etapas, o que minimizou e organizou o tempo de desenvolvimento.

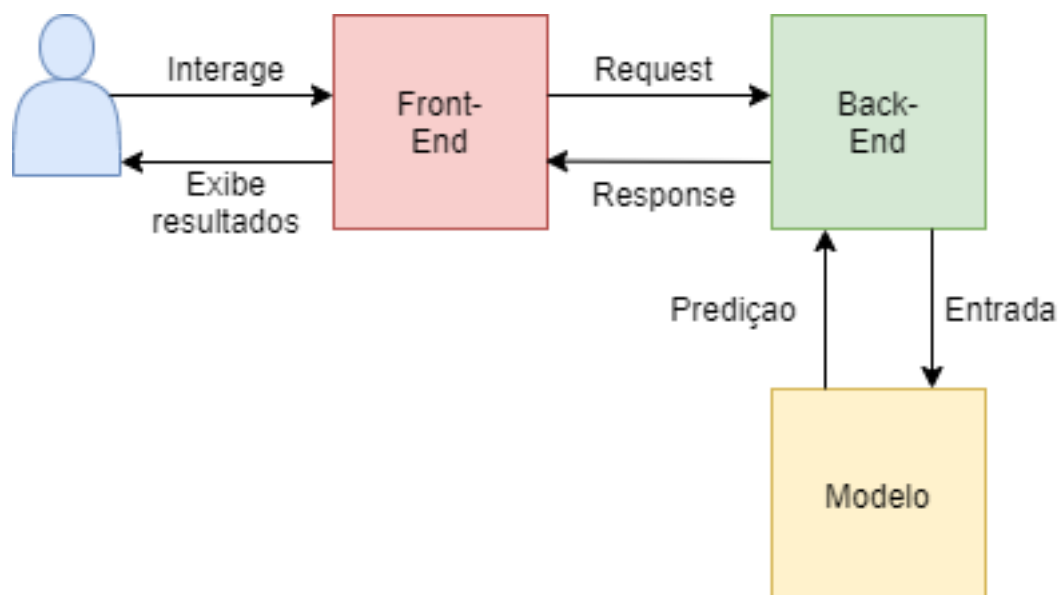


Figura 1 – Arquitetura geral

O Sistema de Predição de Resultados de Apostas Esportivas é composto por três camadas: *Front-end*, *Back-end*, e Modelo, como apresentado no diagrama acima.

Desse modo se no futuro, for necessário substituir a camada de *front-end*, por exemplo, basta remove-la que os outros dois módulos continuarão funcionando de modo independente. Assim como se for necessário alterar o modelo preditivo, basta substituí-lo que tanto o *front-end* quanto o *back-end* continuarão funcionando.

A camada *front-end* é a responsável por exibir as informações em um formato mais compreensível para os usuários. Nela são feitas requisições para a camada *back-end*, que ficará responsável por preparar os dados e entrega-los como entrada para a camada Modelo, a partir disso, o mesmo fará uma predição em cima desses dados e retornará ao *back-end*, que ao receber, retornará para o *front-end* exibir para o usuário.

### 3.2.1 Camada Front-end

Como dito anteriormente a camada *front-end* é a responsável por exibir as informações em um formato mais compreensível para os usuários. Para fazer isso da maneira mais sustentável e escalável utilizamos um conceito parecido com o utilizado na arquitetura geral da aplicação, de modo que a camada *front-end* também foi separada em módulos separados

com funcionalidades específicas.

De forma mais geral a figura 2 mostra como ficou configurada as camadas do *front-end*, onde é tentado agrupar funcionalidades semelhantes em um único módulo e desacoplando-os o máximo possível de outro módulo. De modo que eles possam se comunicar através de uma camada chamada "Camada de aplicação", que fica responsável pela lógica do sistema.

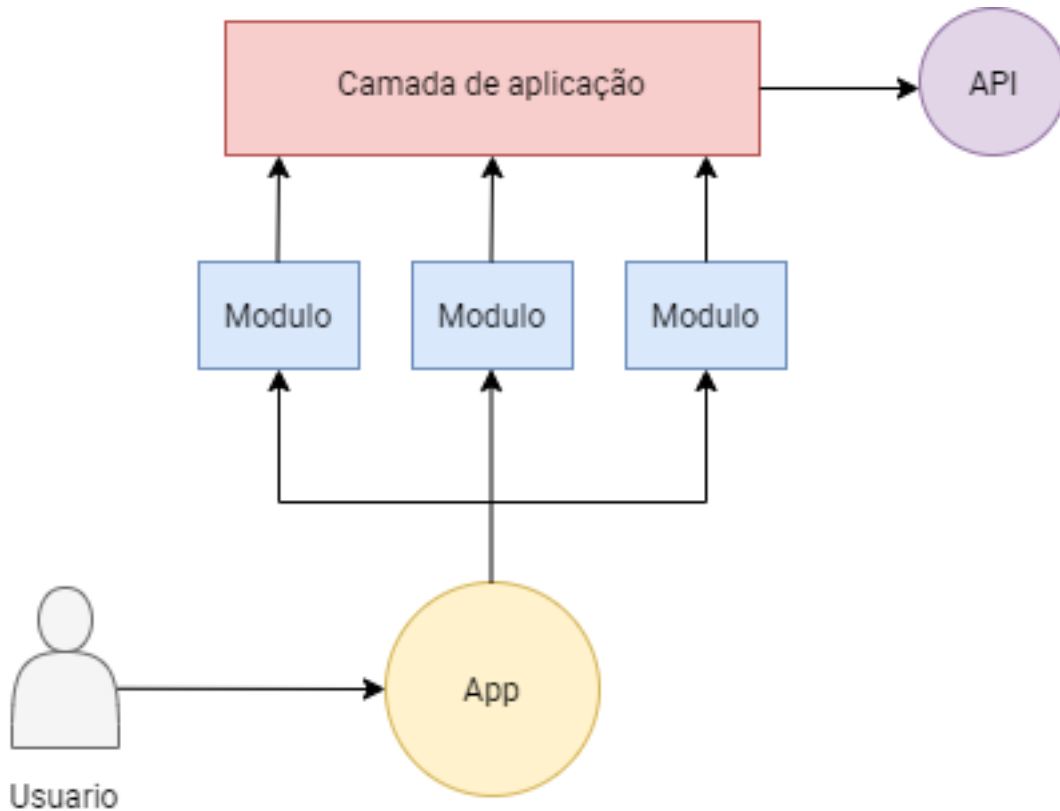


Figura 2 – Arquitetura Geral front-end

Já a figura 3 mostra de forma mais detalhada como os módulos e a camada de aplicação funcionam. Quando o usuário interage com a aplicação, ele é direcionado para um módulo específico através das rotas, cada módulo é composto por Páginas, sendo ela montada por Componentes com funcionalidades específicas, note que os módulos são encapsulados, porém determinados Componentes podem ser reutilizados em outros módulos.

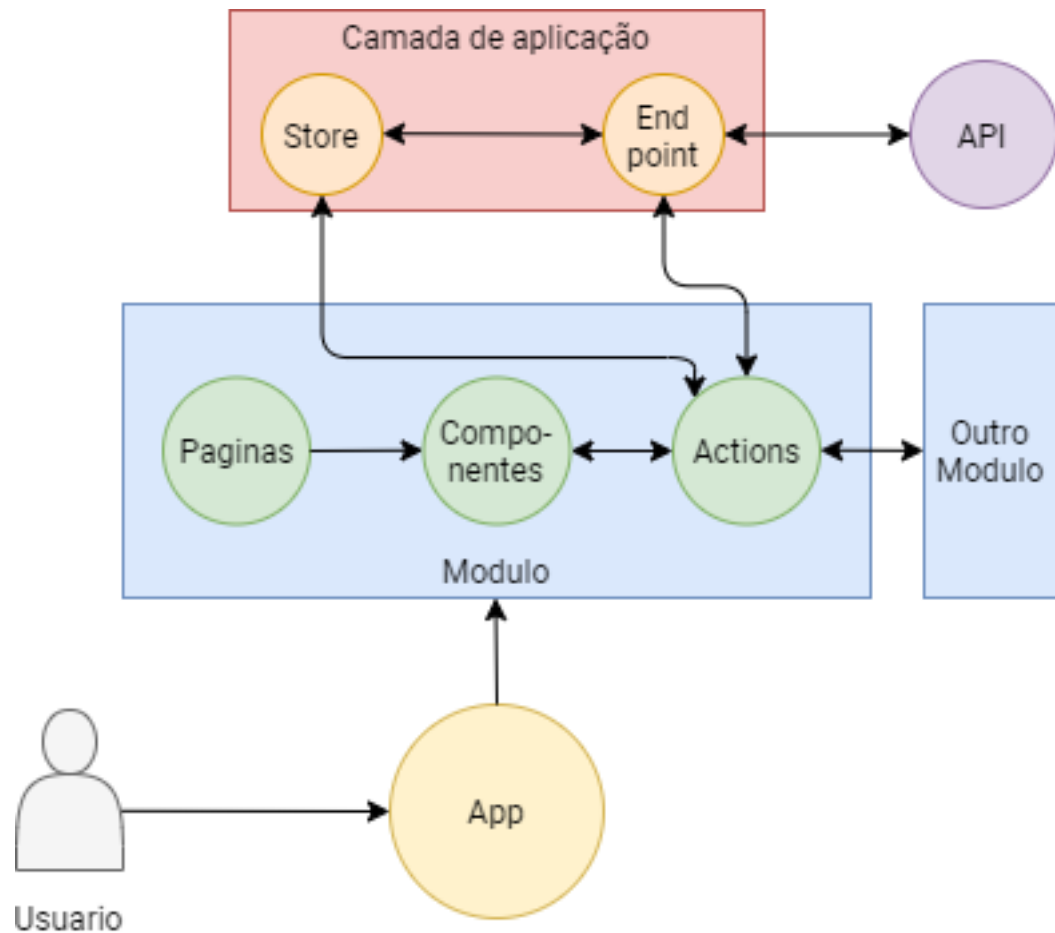


Figura 3 – Arquitetura Detalhada front-end

Apesar dos módulos serem encapsulados, a aplicação em si é completamente interativa, isso se deve ao fato de que cada módulo possui *Actions*, no qual são responsáveis por se comunicar com outros outros módulos, e com a Camada de aplicação.

A Camada de aplicação funcionaria como o núcleo da aplicação, e seria separada em duas partes, uma *Store* e os *Endpoints*. A *Store* é o estado global da aplicação, nela ficará guardados todo o conteúdo, sendo acessível pelos diferentes módulos ao mesmo tempo.

Em alguns momentos, determinadas informações precisam ser requisitadas pela API, então os *Endpoints* atuam como um canal de comunicação entre a aplicação e o *back-end*, e então a receber a response, distribui a mesma para a aplicação.

### 3.2.2 Camada Back-end

A Camada *back-end* é a parte da aplicação que atua no momento em que o *front-end* faz uma requisição através de um *endpoint*, como explicado anteriormente na subseção 3.2.1. A figura 4 abaixo mostra como ficou configurada essa camada.

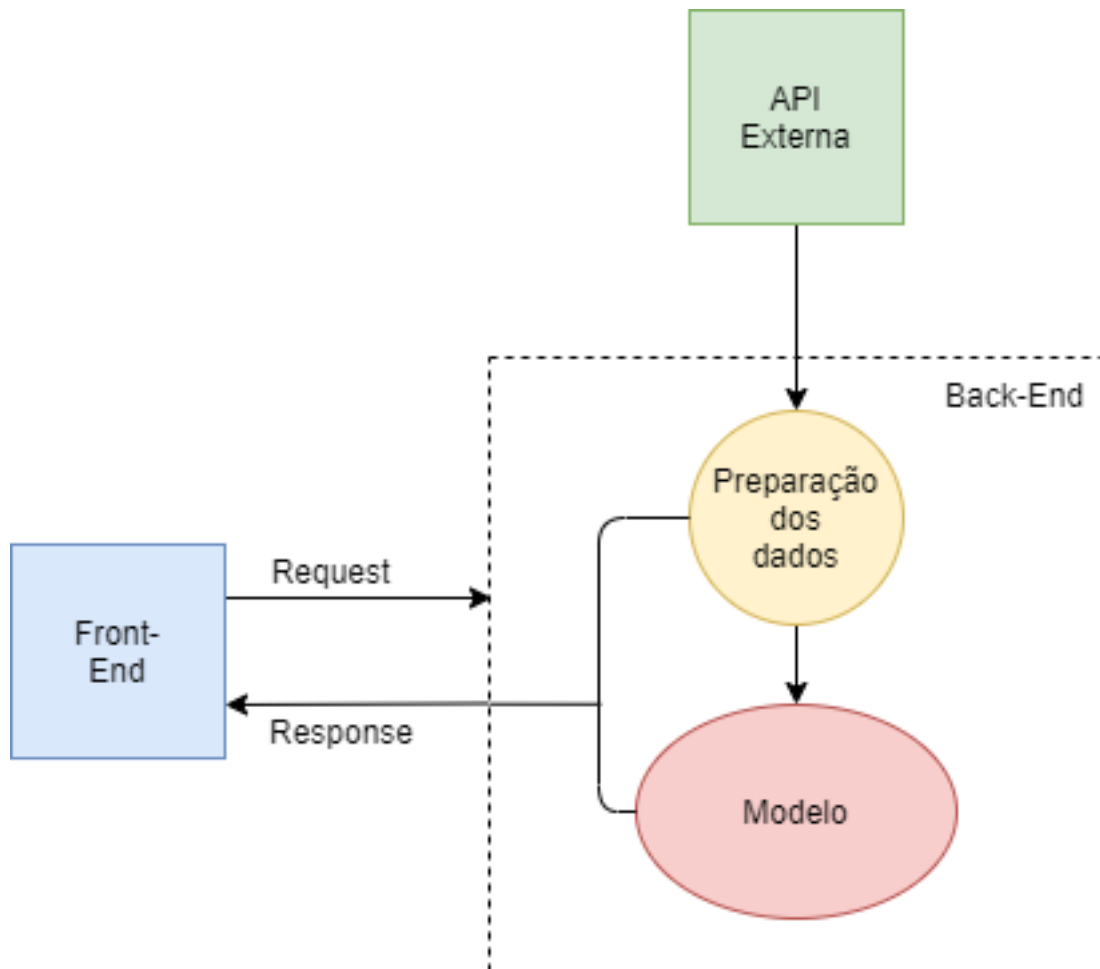


Figura 4 – Arquitetura back-end

Essa camada tem como responsabilidade interligar informações em diferentes locais, onde a partir de um *endpoint*, uma determinada informação é requisitada a uma API externa, porém essa informação ainda não está pronta para ser exibida para o usuário, nem servida como entrada para o modelo, então o *back-end* faz uma serie de preparações para adapta-las a necessidade da nossa aplicação.

Com os dados preparados, agora eles podem ser retornados para o *front-end* para serem exibidos, ou servirem como entradas para a camada Modelo, que retornará uma predição para ser exibida no *front-end*.

### 3.2.3 Camada Modelo

A Camada Modelo é a responsável por realizar a predição de uma determinada partida. Porém existem uma série de etapas para chegar a esse resultado, a figura 5 abaixo mostra de forma mais simplificada como ocorre esse processo.



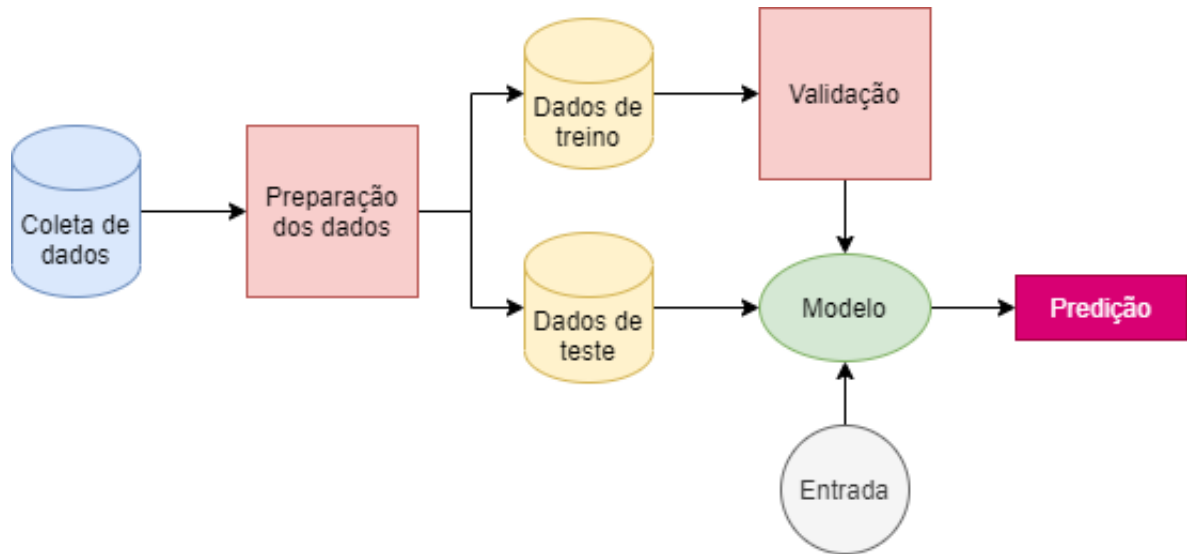


Figura 5 – Arquitetura Modelo

De forma mais geral, para conseguirmos construir um modelo preditivo, o primeiro passo é a coleta de dados, a partir dela, os mesmos serão preparados para o contexto do nosso problema.

Após a preparação dos dados, estes serão divididos em dados de treino e teste, os dados de teste serão utilizados posteriormente para de fato testar a precisão e o desempenho do modelo. Os dados de treino servirão para treinar o algoritmo e ensiná-los a usar esses padrões para resolver problemas, esses dados passarão por um processo de validação, a figura 6 abaixo elucida esse processo.

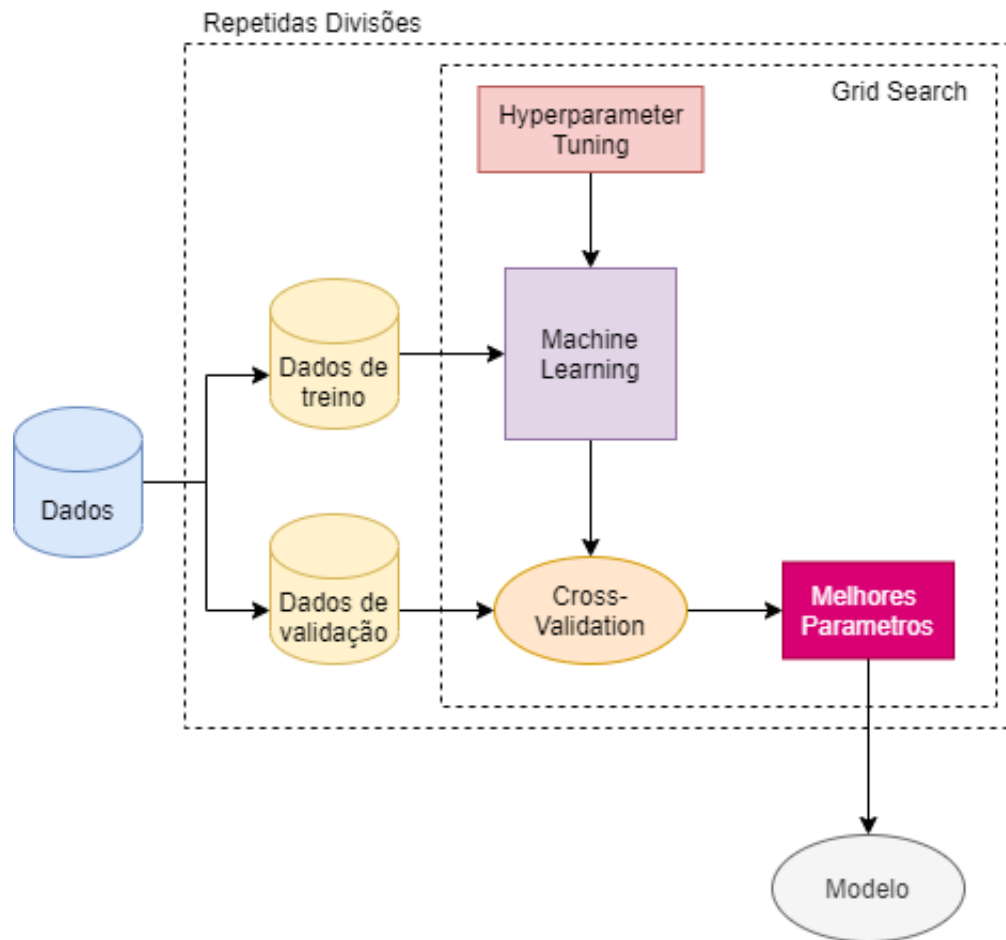


Figura 6 – Arquitetura Modelo Validação

Para avaliarmos cuidadosamente o desempenho do nosso modelo, os nossos dados passarão por um processo chamado validação cruzada aninhada. Esse processo consiste em ao invés de dividir os dados em conjuntos de treinamento e validação uma vez, criamos um *loop* externo que divide repetidamente os dados em conjuntos de treinamento e validação, onde em cada *loop*, ocorre alguns ajustes de parâmetros específicos do modelo. E com esses parâmetros o modelo é treinado e validado. Esse processo é repetido até que um nível de precisão aceitável seja alcançado.

Com os dados devidamente validados e testados, temos um modelo preditivo. Então precisamos de alguma forma interagir com nosso modelo e dar a ele problemas para resolver. No nosso caso, isso acontece através uma API, onde o *back-end* do sistema ficará responsável por passar dados de entrada para o modelo.

### 3.3 Telas do Sistema e Suas Funcionalidades

Como introduzido anteriormente neste capítulo, o Sistema de Predição de Resultados de Apostas Esportivas é uma Aplicação Web que possibilita ao usuário visualizar previsões de

resultados de partidas profissionais de *League of Legends*. Nesta seção iremos apresentar as principais funcionalidades e características do sistema.

### 3.3.1 Tela Horários

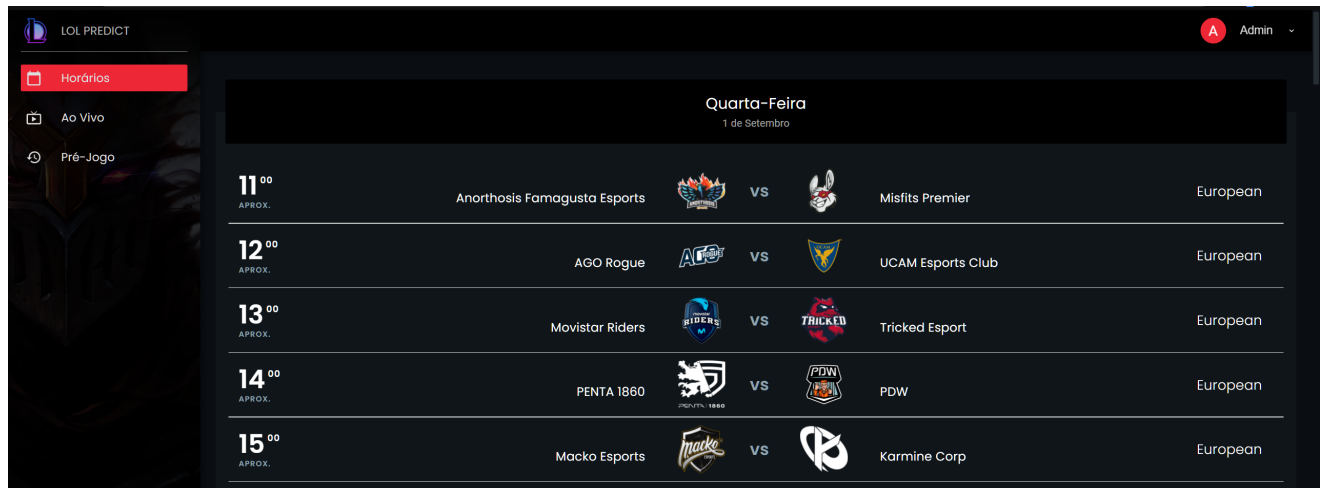


Figura 7 – Interface Horários

A interface horários (Figura 7) pode ser acessada através de um *sidebar*, pressionando o botão 'Horários', esta tela fornece uma lista de partidas que virão a acontecer. Essas partidas estão agrupadas em tabelas pelos seus respectivos dias que serão realizadas. O usuário poderá percorrer a página para visualizar diferentes dias em ordem cronológica. Cada linha dessas tabelas contém o horário aproximado que a partida acontecerá, os times com sua logo e nome, e também o campeonato que estão disputando.

### 3.3.2 Tela Pré-Jogo

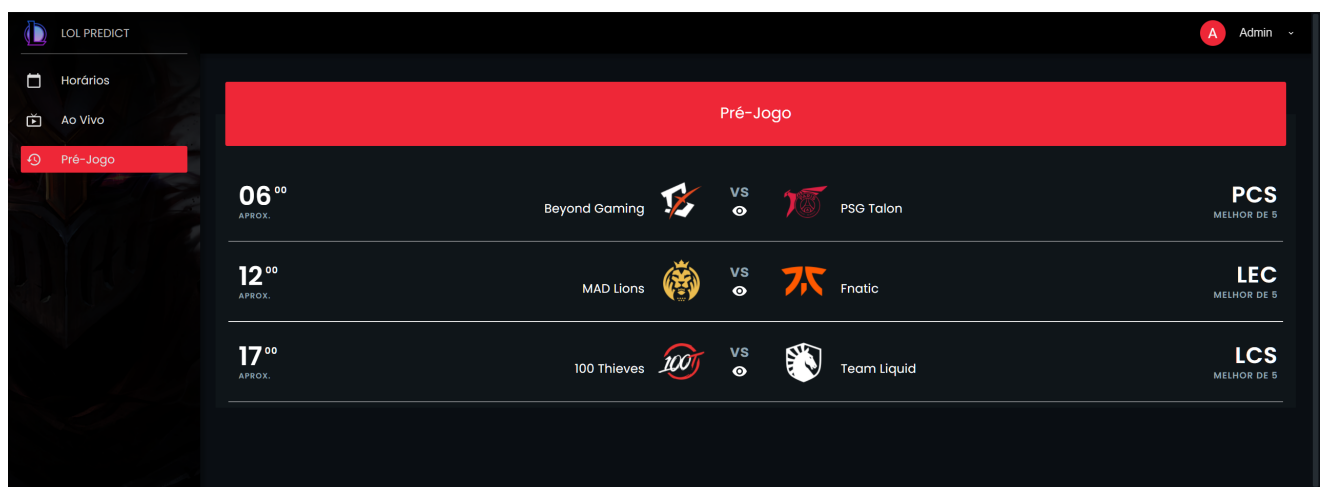


Figura 8 – Interface Pré-Jogo

A interface pré-jogo (Figura 8) pode ser acessada através de um *sidebar*, pressionando o botão 'Pré-Jogo', esta tela fornece a lista de partidas que irão acontecer no dia. Essas partidas estão agrupadas em uma única tabela. O usuário poderá percorrer a página para visualizar as diferentes partidas em ordem cronológica. Cada linha dessas tabelas contém o horário aproximado que a partida acontecerá, os times com sua logo e nome, o campeonato que estão disputando junto com o modelo de disputa da partida, por exemplo melhor de 5, e também um botão 'Visualizar', que levará o usuário para a *live streaming* que a partida será transmitida.

Ao clicar sobre uma partida, ou seja, uma linha da tabela, o usuário poderá visualizar com detalhes diversas estatísticas sobre os times que se enfrentarão, como podemos ver na figura 9 abaixo











12 <sup>00</sup> APROX.			MAD Lions		VS	Fnatic		LEC MELHOR DE 5	
PREDIÇÕES			SPLIT			ANO			
	BLUE	RED	MAD 6-1 <span>U U U U U U U</span>			<span>U U U U U U U</span> 12-6 FNC			
VENCEDOR			-0.11	86%	1.27	1.33	67%	1.61	
	MAD	FNC	GOLD RATE	WIN RATE	KILL/DEATH RATIO	GOLD RATE	WIN RATE	KILL/DEATH RATIO	
TORRE			55.1	30.7	0.95	66.9	-0.3	0.97	
	MAD	FNC	EARLY GAME RATE	MID-LATE RATE	TEAM/OPP KILLS	EARLY GAME RATE	MID-LATE RATE	TEAM/OPP KILLS	
BARÃO			57%	57%	43%	72%	72%	61%	
	MAD	FNC	FIRST BLOOD	FIRST DRAGON	FIRST TOWER	FIRST BLOOD	FIRST DRAGON	FIRST TOWER	
DRAGÃO			49.6%	50.7%	57%	49.2%	53.9%	72%	
	MAD	FNC	LANE CONTROL	JUNGLE CONTROL	FIRST THREE TOWERS	LANE CONTROL	JUNGLE CONTROL	FIRST THREE TOWERS	
OVER			57%	50%	50%	68%	70%	66%	
	MAD	FNC	BARON RATE	DRAGON RATE	HAROLD RATE	BARON RATE	DRAGON RATE	HAROLD RATE	

Figura 9 – Componente Pré-Jogo - Split

O componente está separado em duas partes: Predições, e Estatísticas. A primeira parte será a responsável por apresentar as predições das partidas pré-jogo para diferentes mercados de aposta, essa funcionalidade será discutida posteriormente na seção 5.2, porém o *layout* da aplicação web foi projetado para receber essas predições.

A segunda parte do componente são as estatísticas, onde esses dados podem ser referentes as estatísticas do Split (Temporada), ou ao Ano, com o usuário podendo selecionar o modo através das *tabs* ( SPLIT | ANO ). A partir disso, temos dois lados que exibem os dados dos times, sendo o lado esquerdo referente ao time Azul (Figura 10), e o lado direito referente ao time Vermelho (Figura 11).

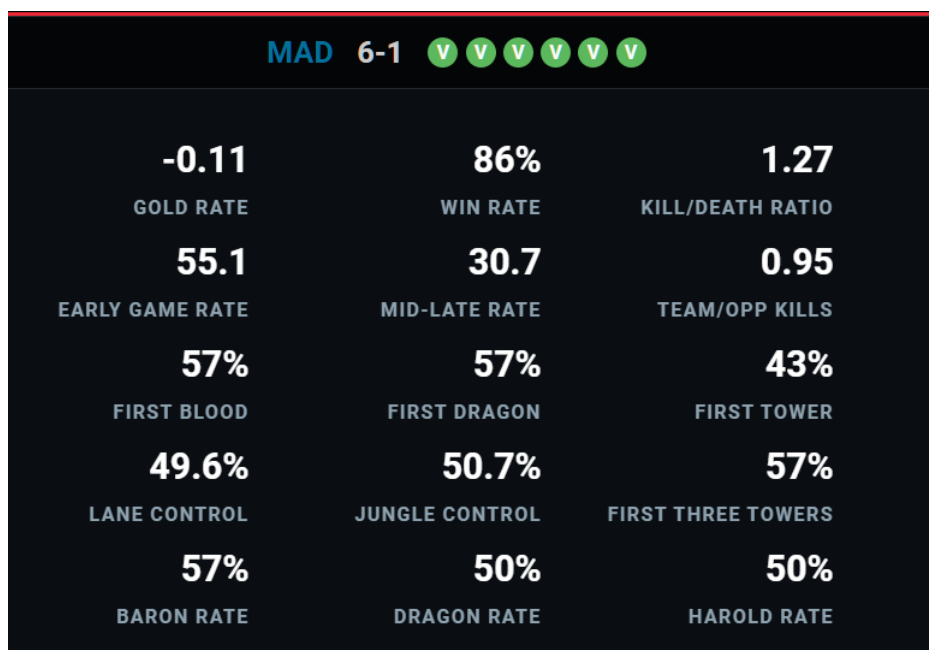


Figura 10 – Componente Estatísticas Time Azul

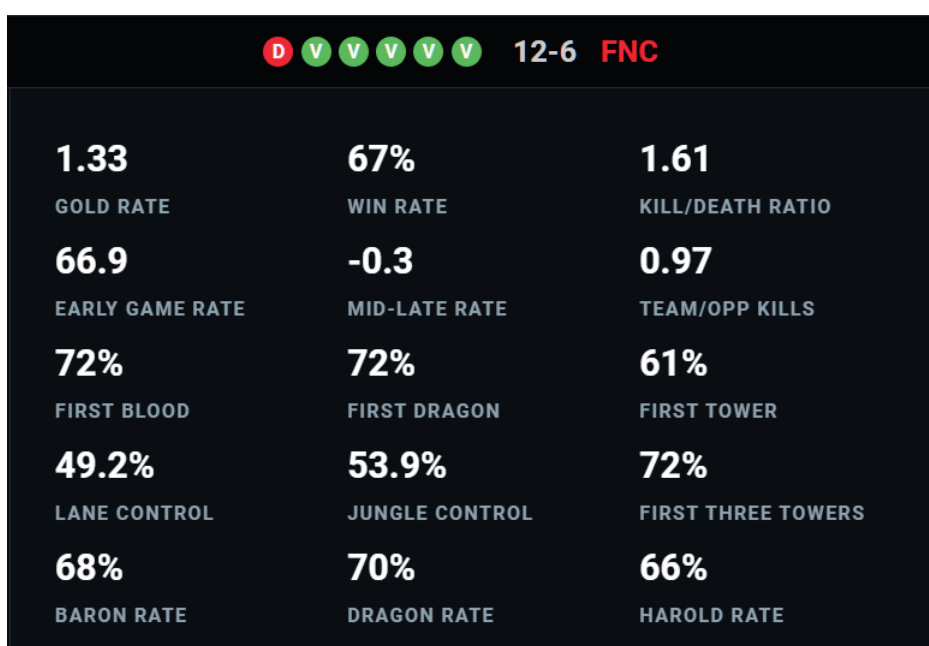


Figura 11 – Componente Estatísticas Time Vermelho

### 3.3.3 Tela Ao Vivo

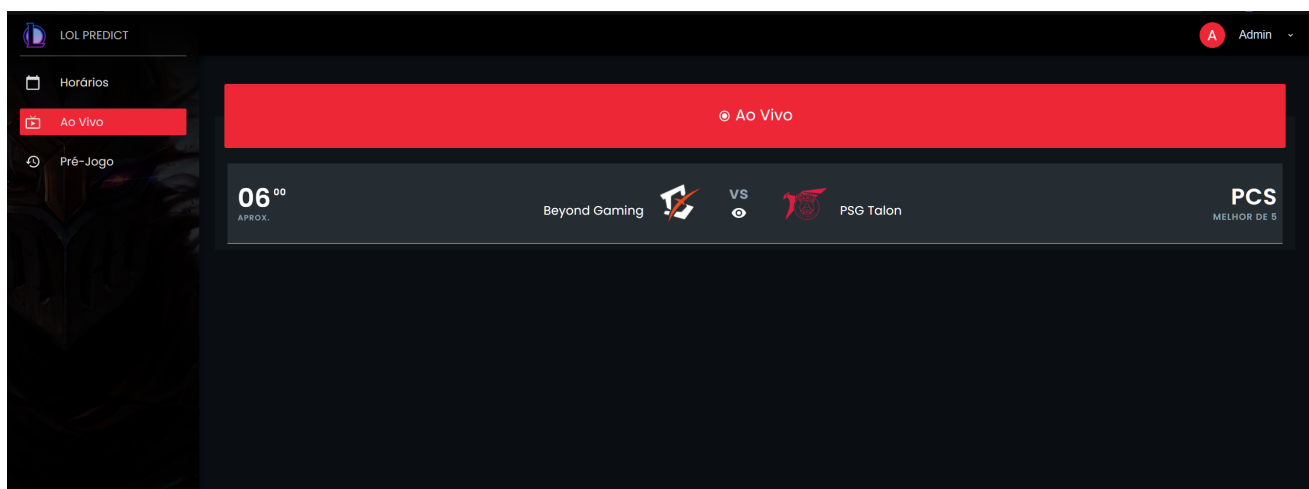


Figura 12 – Interface Ao Vivo

Por fim temos a interface Ao Vivo (Figura 12), ela possui um *layout* relativamente bem parecido com a interface Pré-jogo 3.3.2, ela pode ser acessada através de um *sidebar*, pressionando o botão 'Ao Vivo', e nela estão apresentados a lista de partidas que estão acontecendo no momento. Essas partidas estão agrupadas em uma única tabela, e o usuário poderá interagir com as partidas (linha da tabela) da mesma forma que foi explicado na seção anterior 3.3.2.

Ao clicar sobre uma partida, o usuário poderá visualizar os detalhes da partida que está acontecendo, como podemos ver na figura 13 abaixo.

06:00 APROX.	Beyond Gaming		VS	PSG Talon		PCS MELHOR DE 5
PREDIÇÕES 15		ESTATÍSTICA			HISTÓRICO	
	BLUE	RED	PARTIDA 5	PSG 10-5	BYG 8-5	31:28
VENCEDOR	PSG	BYG	60171 OURO RECEBIDO	18 ABATES	1 ABATES	48286 OURO RECEBIDO
TORRE	PSG	BYG	68860 XP RECEBIDO	1064 TROPAS ABATIDAS	1061 TROPAS ABATIDAS	59080 XP RECEBIDO
BARÃO	PSG	BYG	3 DRAGÕES REALIZADOS	1 BARONS REALIZADOS	0 BARONS REALIZADOS	2 DRAGÕES REALIZADOS
DRAGÃO	PSG	BYG	2 INIBIDORES DESTRUÍDOS	11 TORRES DESTRUÍDAS	1 TORRES DESTRUÍDAS	0 INIBIDORES DESTRUÍDOS
OVER	PSG	BYG				

Figura 13 – Componente Ao Vivo Estatísticas

Este componente também está separado em duas partes: Predições, e Estatísticas. A primeira parte é a responsável por apresentar as predições da partida para diferentes mercados de aposta, no momento somente a predição do vencedor da partida está implementada, as outras funcionalidades serão discutidas posteriormente na seção 5.2, porém o *layout* já está preparado para receber essas predições.

A predição do vencedor da partida acontece em dois momentos, ao chegar aos 10 minutos, o nosso modelo preditivo retornará uma predição, e ela será exibida para o usuário através de uma marcação verde no time predito (Figura 14). Em um segundo momento, quando a partida chegar aos 15 minutos, o modelo preditivo automaticamente retornará uma predição, e a marcação será atualizada para o novo resultado (Figura 15).









PREDIÇÕES <u>15</u>		
	BLUE	RED
VENCEDOR	 PSG	 BYG
TORRE	 PSG	 BYG
BARÃO	 PSG	 BYG
DRAGÃO	 PSG	 BYG
OVER	 PSG	 BYG

Figura 14 – Componente Predição 10 Minutos











PREDIÇÕES <u>15</u>		
	BLUE	RED
VENCEDOR	 PSG	 BYG
TORRE	 PSG	 BYG
BARÃO	 PSG	 BYG
DRAGÃO	 PSG	 BYG
OVER	 PSG	 BYG

Figura 15 – Componente Predição 15 Minutos

A segunda parte do componente são as estatísticas, onde esses dados podem ser referentes as estatísticas em tempo real, ou as estatísticas referentes aos jogos passados, como aquelas exibidas na figura 9. O usuário poderá selecionar qual modo deseja através das *tabs* ( ESTATÍSTICA | HISTÓRICO ). A partir disso, temos dois lados, sendo o lado esquerdo referente aos dados time Azul, juntamente com qual partida está sendo disputada, como por exemplo, partida 5 da melhor de 5 (Figura 16), e o lado direito referente aos dados do time Vermelho, juntamente com o tempo exato da partida (Figura 17).



Figura 16 – Componente Tempo Real Time Azul



Figura 17 – Componente Tempo Real Time Vermelho



## 4 Avaliação da Proposta

Este capítulo apresenta a avaliação do modelo de predição. A avaliação tem como objetivo verificar o desempenho das predições, ou seja, se os usuários estão sendo recomendados com os resultados corretos dos vencedores da partida

Este capítulo está organizado da seguinte forma: Seção 4.1 detalha sobre o *dataset* que foi utilizado no projeto e a sua preparação. Seção 4.2 explica o método de teste e as métricas utilizadas como base na avaliação e otimização, assim como apresenta os diferentes modelos de classificação que foram treinados e suas comparações. E por fim a Seção 4.3 apresenta o método utilizado para realizar as otimizações nos modelos, exibindo os resultados obtidos, assim como o modelo final escolhido.

### 4.1 Conjunto de Dados

Para prever os resultados das partidas, utilizaremos dados da temporada atual das competições oficiais de *League of Legends*. O conjunto de dados foi obtido através do [Oracle](#), no qual é disponibilizado gratuitamente e destina-se ao uso por analistas, comentaristas e fãs. No qual inclui as seguintes informações:

- Dados de mais de 40000 partidas profissionais.
- Estatísticas históricas de mais de 2400 times.
- Escalações dos times
- Temporadas 2014 até 2021
- Todas as principais ligas (LCS, LEC, LCK, LPL, PCS, CBLol, ...)
- Detalhes relevantes das partidas (*gold*, *xp*, *creeps*, *kills* e *assists* de ambos os times)
- Detalhes relevantes das partidas (*gold*, *xp*, *creeps*, *kills* e *assists* de ambos os times) aos 10 minutos e aos 15 minutos
- Estatísticas históricas relevantes ( *Win-Rate* , *Gold-Rate* , *Early-Game-Rate* , *Mid-Late-Game-Rate* )

#### 4.1.1 Descrição

Para a construção conjunto de dados utilizado no modelo, foi utilizado unir dois tipos de dados, o primeiro seriam os dados de uma partida que foi realizada, e o segundo seriam

os dados referentes aos históricos dos times que eles possuíam na data em que a partida foi disputada.

Este processo aconteceu a partir dos dados disponibilizados através do [Oracle](#), onde os dados das partidas são disponibilizados por um csv. Os dados referentes aos históricos dos times foram construídos, a partir de um *endpoint* também disponibilizado pelo mesmo, onde é possível recuperar estatísticas históricas de todos os times.

Então se utilizando do mesmo, foi criado um conjunto de dados onde cada time que disputou uma competição oficial na temporada atual, em determinadas datas disponíveis na API, possui os seus dados históricos.

Desta forma, com essas informações em mãos foi feita a união dos dados da partida com os dados históricos dos times que eles possuíam na data em que a partida foi disputada.

Para o nosso modelo, decidimos utilizar somente dados desta temporada 2021, pois diferente de esportes mais tradicionais, as 'regras' do jogo mudam bastante de uma temporada para outra e dados passados não são tão correlatos com resultados. Isso nos dá um conjunto de dados utilizável de:

- Mais de 8000 partidas profissionais de todas as ligas.
- Estatísticas históricas de mais de 300 times.

#### 4.1.2 Preparação

Uma etapa importante antes de construir nosso modelo é analisar e preparar os dados para ter certeza de que está adequado para usarmos ao treinar e testar modelos diferentes.

Três etapas de preparação foram realizadas para conseguir isso:

- Como explicado anteriormente na seção [4.1.1](#) as estatísticas históricas dos times foram obtidas através de um processo de junção com os dados das partidas. O primeiro passo para conseguir isso foi obter a lista de todos os times disponíveis, a partir disso são feitas requisições para obter as estatísticas e então é montada uma tabela com as informações de todos os times. Após isso, então nós unimos as duas tabelas utilizando o nome dos times que disputam a partida com suas respectivas estatísticas históricas na data correspondente.
- As partidas disputadas durante a temporada costumam ser realizadas em diferentes atualizações do jogo, no qual as 'regras' do jogo são alteradas. Para ter informações mais correlatas com o resultado, os dados são filtrados para somente partidas jogadas em atualizações recentes, por exemplo, se estamos na atualização 11.9, serão utilizadas somente partidas a partir da atualização 11.6.

- Algumas partidas do conjunto de dados não possuem todos atributos necessários para predição, nesse caso a respectiva linha da tabela será deletada.
- Algumas variáveis do conjunto de dados não possuem relevância para realizar as nossas predições, como por exemplo, os nomes dos times, nomes dos jogadores. Então foram selecionadas somente as variáveis necessárias

### 4.1.3 Variáveis

Nesta subseção serão apresentadas as variáveis selecionadas para usar nos nossos modelos. Para entender melhor o significado e a importância faremos uma explicação sobre as mesmas.

*Gold* é a moeda do jogo em League of Legends. É usado para comprar itens na loja que fornecem aos jogadores *status* melhorados e habilidades de bônus, o que, por sua vez, é uma das principais maneiras de os campeões aumentarem seu poder no decorrer de um jogo.

Experiência de campeão (XP) é uma mecânica de jogo que permite que os jogadores subam de nível depois de atingir certa quantidade de experiência. Subindo de nível permitirá a eles acessarem novas habilidades ou níveis mais altos de habilidades existentes. Muitas *status* básicos e alguns itens e runas se adaptam ao nível do campeão.

*Creeps* são unidades que constituem a principal força enviada pelo Nexus. Eles surgem periodicamente de seu Nexus e avançam ao longo das rotas em direção ao Nexus inimigo, enfrentando automaticamente qualquer unidade ou estrutura inimiga que encontrarem. Eles são controlados por inteligência artificial e usam apenas ataques básicos. Ao abate-los, concederão aos jogadores XP e *gold*

Uma *kill* é o evento que reduz a vida de um jogador inimigo a zero e faz com que ele entre no estado de morte com sucesso, fazendo com que o jogador fique impossibilitado de participar da partida por um tempo determinado dependendo da duração da partida. Quando um jogador é eliminado, todos os jogadores que contribuíram para a morte recebem XP e *gold*, sendo que o jogador que deu o último golpe no inimigo receberá mais *gold* por isso.

*Assist* é a ação de ajudar um campeão aliado a matar um campeão inimigo. Quando um campeão é morto, o assassino recebe uma certa quantidade de *gold*, enquanto uma quantidade reduzida é compartilhada entre todos os campeões que ajudaram na eliminação.

*Towers* são fortificações que atacam unidades inimigas à vista. As torres são um componente central de League of Legends. Elas causam danos aos inimigos e fornecem visão para sua equipe, permitindo-lhes controlar melhor o campo de batalha. As equipes devem destruir torres inimigas para empurrar seu ataque em território inimigo, ao destruí-las concederão aos jogadores uma grande quantidade de *gold* por isso.

As unidades *Dragon*, *Harold* e *Baron* são monstros neutros em League of Legends. Ao

contrário dos *creeps*, os monstros não lutam por nenhum dos times, e só o farão se forem provocados. Ao abate-los concederão aos jogadores de um time vantagens favoráveis em relação ao adversário.

- Variáveis da Partida

- golddiffat15 (Diferença de *gold* entre os times aos 15 minutos)
- xpdiffat15 (Diferença de *XP* entre os times aos 15 minutos)
- csdiffat15 (Diferença de *creeps* entre os times aos 15 minutos)
- killsdiffat15 (Diferença de *kills* entre os times aos 15 minutos)
- assistsdiffat15 (Diferença de *assists* entre os times aos 15 minutos)
- golddiffat10 (Diferença de *gold* entre os times aos 10 minutos)
- xpdiffat10 (Diferença de *xp* entre os times aos 10 minutos)
- csdiffat10 (Diferença de *creeps* entre os times aos 10 minutos)
- killsdiffat10 (Diferença de *kills* entre os times aos 10 minutos)
- assistsdiffat10 (Diferença de *assists* entre os times aos 10 minutos)

- Variáveis das Estatísticas dos Times

- WR (*Win-Rate* do Time Azul)
- KD (*Kill-to-Death Ratio* do Time Azul)
- GPR (*Gold Percent Ratio* do Time Azul)
- GSPD (*Average Gold Spent Ratio* do Time Azul)
- EGR (*Early-Game-Rate* do Time Azul)
- MLR (*Mid-Late-Game-Rate* do Time Azul)
- FB% (*First Blood Rate* do Time Azul)
- FT% (*First Blood Rate* do Time Azul)
- F3T% (*First To Three Towers Rate* do Time Azul)
- HLD% (*Harold Rate* do Time Azul)
- DRG% (*Dragon Rate* do Time Azul)
- BN% (*First Blood Rate* do Time Azul)
- LNE% (*Lane Control Rate* do Time Azul)
- JNG% (*Jungle Control Rate* do Time Azul)
- OPP\_WR (*Win-Rate* do Time Vermelho)
- OPP\_KD (*Kill-to-Death Ratio* do Time Vermelho)

- OPP\_GPR (*gold Percent Ratio*) do Time Vermelho
- OPP\_GSPD (*Average Gold Spent Ratio*) do Time Vermelho
- OPP\_EGR (*Early-Game-Rate*) do Time Vermelho
- OPP\_MLR (*Mid-Late-Game-Rate*) do Time Vermelho
- OPP\_FB% (*First Blood Rate*) do Time Vermelho
- OPP\_FT% (*First Blood Rate*) do Time Vermelho
- OPP\_F3T% (*First To Three Towers Rate*) do Time Vermelho
- OPP\_HLD% (*Harold Rate*) do Time Vermelho
- OPP\_DRG% (*Dragon Rate*) do Time Vermelho
- OPP\_BN% (*First Blood Rate*) do Time Vermelho
- OPP\_LNE% (*Lane Control Rate*) do Time Vermelho
- OPP\_JNG% (*Jungle Control Rate*) do Time Vermelho

## 4.2 Experimentos

Um dos passos para a construção de um modelo de classificação é testá-lo para que saibamos se o modelo é capaz de generalizar bem para novos dados. Nesta seção, definimos o método de teste e métrica que usamos como base para obter o melhor desempenho possível. Após isso será apresentado os experimentos realizados, tal como suas comparações.

### 4.2.1 Repeated k-Fold Cross-Validation

*Cross-Validation* é um método de treinamento de um modelo para evitar *overfitting*, que é a situação em que o modelo se ajusta muito bem aos dados de treinamento, mas não generaliza para dados que não foram vistos antes.

No *k-fold Cross-Validation*, você divide os dados de entrada em subconjuntos de dados  $k$  (também chamados de *folds*). O modelo é treinado em todos, menos em um ( $k-1$ ) dos conjuntos de dados e, em seguida, avalia o modelo no conjunto de dados que não foi usado para treinamento. Esse processo é repetido  $k$  vezes, com um subconjunto diferente reservado para avaliação (e excluído do treinamento) a cada vez.

Porém ainda assim a estimativa do desempenho do modelo por meio do *k-fold Cross-Validation* pode não ser exata. Isso significa que cada vez que o procedimento é executado, uma divisão diferente do conjunto de dados em *k-folds* vai ser feito, e com isso, a estimativa do desempenho pode ser diferente, resultando em uma estimativa média diferente de desempenho do modelo.

Uma estimativa destorcida pode atrapalhar a avaliação, pois pode não deixar claro o desempenho usado para comparar e selecionar um modelo final para resolver o problema. Uma solução para resolver isso e que foi utilizado em nosso trabalho é a utilização do *Repeated k-fold Cross-Validation*, onde o processo de *k-fold Cross-Validation* é repetido várias vezes e ao final do mesmo é feita a média dos desempenhos em todos os *k-fold* e repetições. No nosso caso foi utilizado  $k = 10$  *folds* sendo eles repetidos  $n = 5$  vezes.

#### 4.2.2 Métricas

Para saber se seu modelo é bom ou ruim, é crucial a utilização de métricas apropriadas para determinado problema. O valor delas reflete a qualidade de um modelo, portanto se forem mal escolhidas, será impossível avaliar se o modelo de fato está atendendo os requisitos necessários.

Um modelo de classificação em geral tem como objetivo decidir em qual classe uma nova observação pertence dentre duas classes denominadas de positiva (P) e negativa (N), que indicam a ocorrência ou não de um determinado evento.

A avaliação desse modelo é feita a partir da comparação entre as classes preditas pelo modelo e as classes verdadeiras de cada exemplo. Todas as métricas de classificação têm um objetivo em comum, que é medir quão distante o modelo está da classificação perfeita, porém fazem isto de formas diferentes.

Para o contexto do nosso problema, a mais adequada é a acurácia, já que estamos interessados em tanto se o time azul vencer (1) quanto se o time vermelho vencer (0), e a acurácia nos diz quantos de nossos exemplos foram de fato classificados corretamente, independente da classe. E como os erros não possuem pesos diferentes, apesar de ser uma métrica simples, ela acaba se encaixando perfeitamente.

A formula a seguir descreve como a métrica é definida:  $\frac{TP+TN}{TP+TN+FP+FN}$   
Como podemos ver na formula, está métrica é definida pela razão entre o que o modelo acertou (TP + TN) e todos os exemplos (TP + TN + FP + FN).

#### 4.2.3 Modelos de Classificação

Existem diversos métodos, com abordagens e técnicas distintas para a resolução do problema de classificação, e não há uma regra geral que defina qual é o método mais adequado para um determinado problema.

Nesta seção, veremos os experimentos realizados para determinar quais abordagens demonstram maiores potenciais para se tornar o modelo final. Portanto a princípio iremos testar e comparar 4 modelos de classificação, que são:

- *k-Nearest-Neighbors*

- *Gaussian Naive Bayess*
- *Logistic Regression*
- *Random Forest Classifier*

#### 4.2.4 k-Nearest-Neighbors

O algoritmo *k-Nearest-Neighbors*, tem como ideia principal determinar o rótulo de classificação de uma amostra baseado nas amostras vizinhas advindas de um conjunto de treinamento. Funcionando da seguinte forma, dada uma instância de teste  $x_q$ , o algoritmo encontra os  $k$  vizinhos mais próximos de  $x_q$  no conjunto de treinamento. Em seguida, a classe de  $x_q$  é dada pela classe que ocorrer com maior frequência entre os  $k$  vizinhos.

Para o nosso experimento, iremos utilizar o algoritmo **KNeighborsClassifier** disponível pela biblioteca [scikit-learn](#), com os seguintes parâmetros *default*:

- `n_neighbors`: 5
- `weights`: 'uniform'
- `algorithm`: 'auto'
- `leaf_size`: 30
- `p`: 2
- `metric`: 'minkowski'
- `metric_params`: None
- `n_jobs`: None

Os resultados que nós obtivemos após o processo de validação 4.2.1 será apresentado na figura 18 abaixo. As acurácias apresentadas são a média de todas acurácias do processo de *cross-validation*

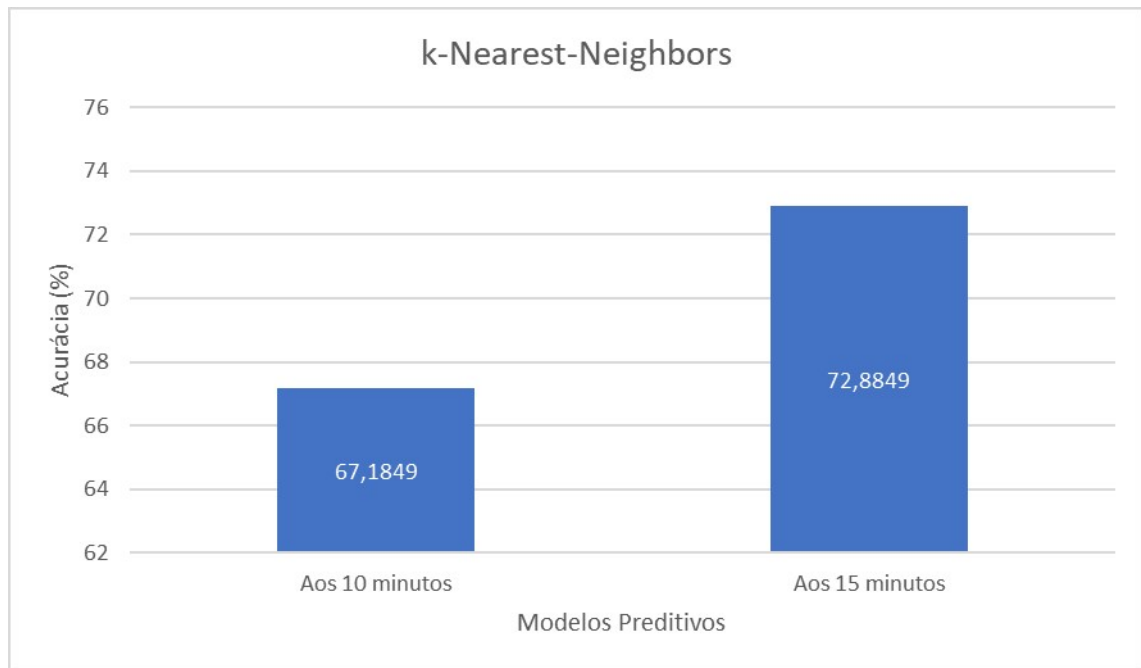


Figura 18 – Resultados kNN

Com a finalidade de compreender melhor o desempenho preditivo do nosso modelo, criamos algumas variantes do mesmo, são elas:

- Predição Pré-Jogo
- Predição aos 10 minutos utilizando somente dados da partida
- Predição aos 15 minutos utilizando somente dados da partida

Os resultados que nós obtivemos após o mesmo processo de validação [4.2.1](#) realizado anteriormente será apresentado na figura [19](#) a seguir:



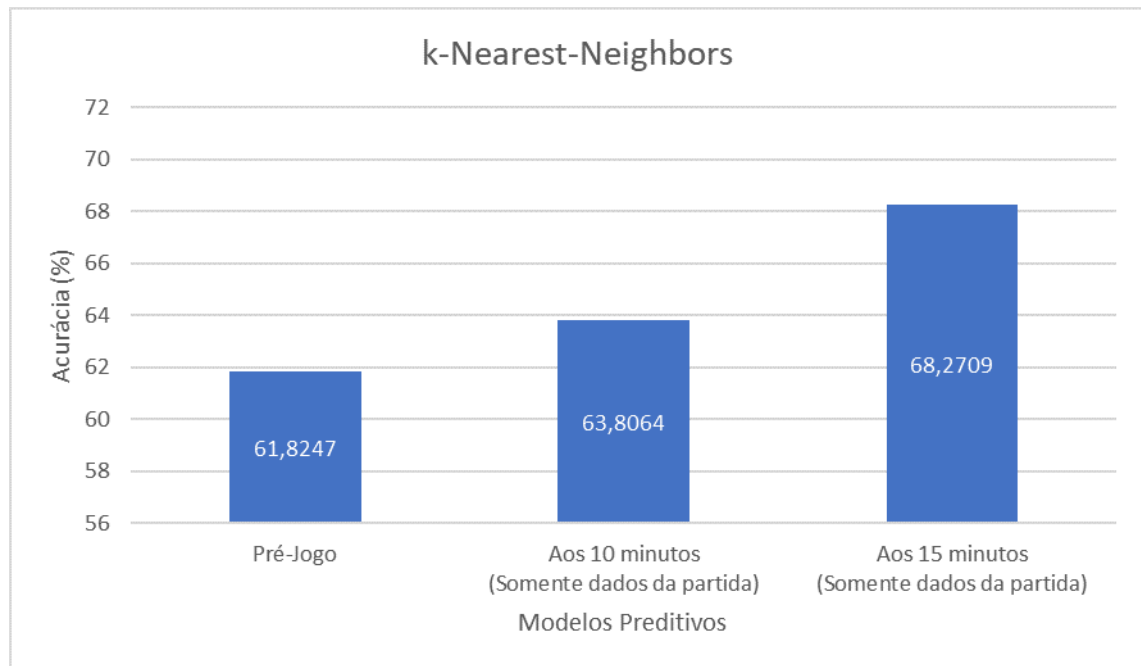


Figura 19 – Resultados Variantes kNN

#### 4.2.5 Naive Bayes

*Naive Bayes* é um algoritmo inspirado no “Teorema de Bayes”, e baseia-se na probabilidade de cada evento ocorrer, desconsiderando a correlação entre as variáveis. Por ter uma matemática relativamente simples, precisa de poucos aperfeiçoamentos para ter uma boa acurácia.

Para o nosso experimento, iremos utilizar o algoritmo **GaussianNB** disponível pela biblioteca [scikit-learn](#), com os seguintes parâmetros *default*:

- priors: None
- var\_smoothing: 1e-09

Os resultados que nós obtivemos após o processo de validação 4.2.1 será apresentado na figura 28 abaixo. As acurácias apresentadas são a média de todas acurácias do processo de *cross-validation*.

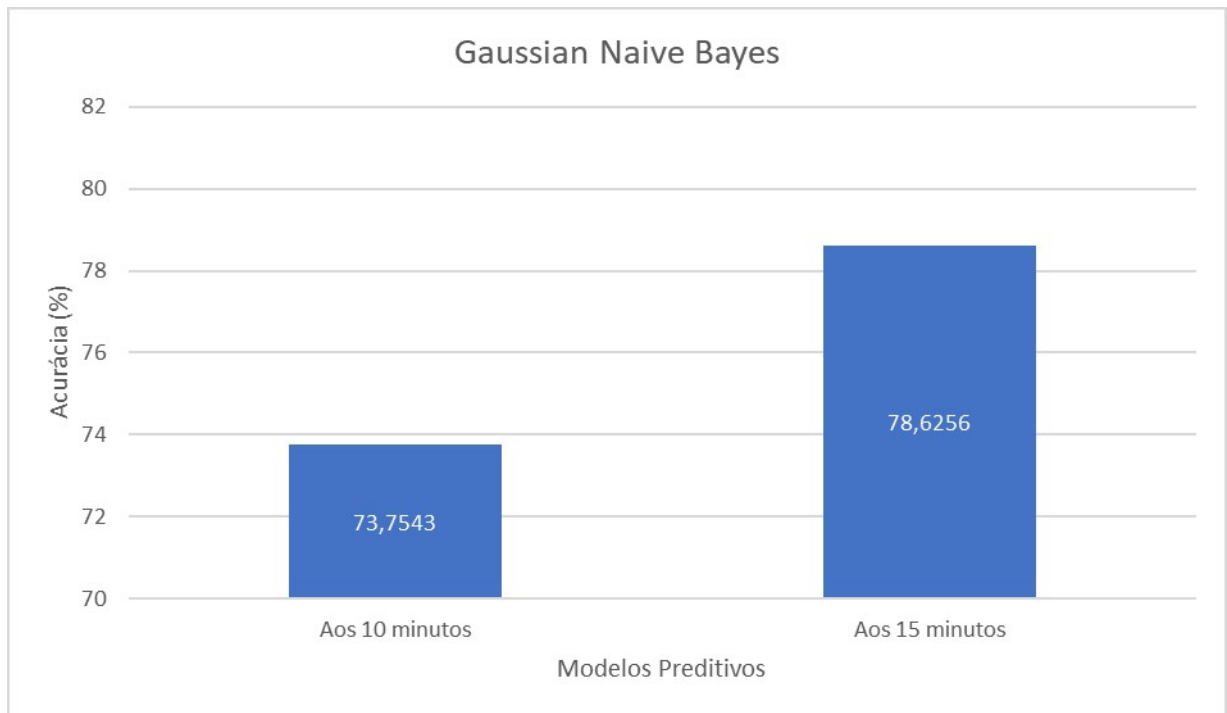


Figura 20 – Resultados Naive Bayes

Com a finalidade de compreender melhor o desempenho preditivo do nosso modelo, criamos algumas variantes do mesmo, são elas:

- Predição Pré-Jogo
- Predição aos 10 minutos utilizando somente dados da partida
- Predição aos 15 minutos utilizando somente dados da partida

Os resultados que nós obtivemos após o mesmo processo de validação [4.2.1](#) realizado anteriormente será apresentado na figura [21](#) abaixo. As acurácias apresentadas são a média de todas acurácias do processo de *cross-validation*.

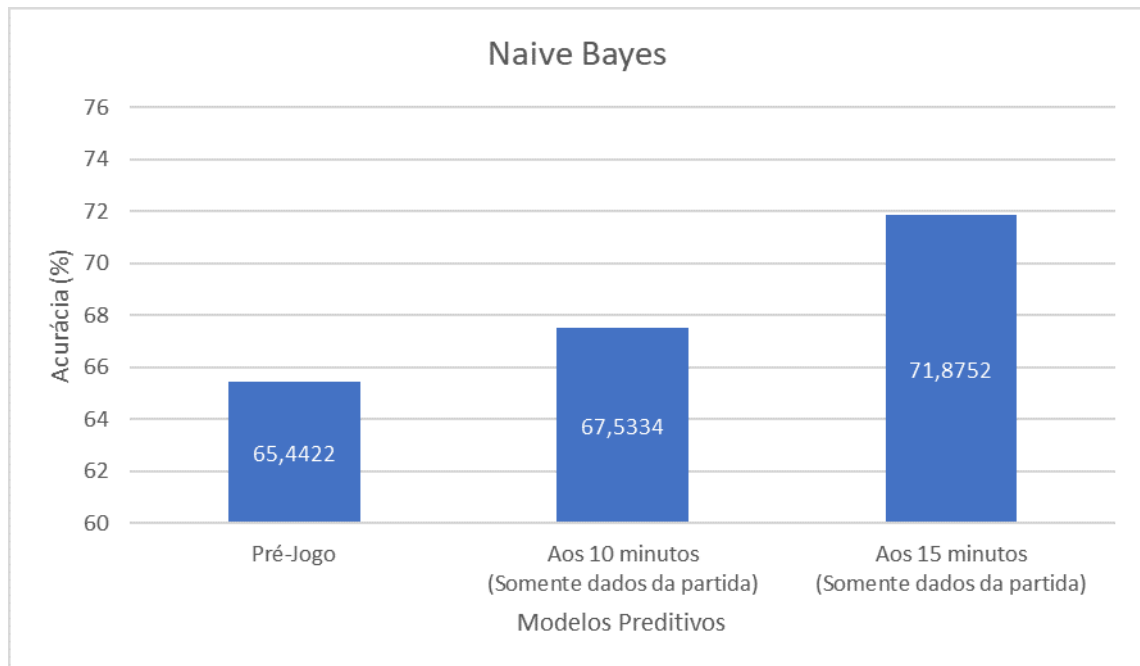


Figura 21 – Resultados Variantes Naive Bayes

#### 4.2.6 Logistic Regression

O objetivo da *Logistic Regression* é similar ao da *Linear Regression*, produzir um modelo linear que permite estimar a probabilidade associada à ocorrência de um dado exemplo pertencer a uma classe (positiva ou negativa) dado um conjunto de atributos. Porém, diferentemente do modelo produzido pela *Linear Regression*, o modelo logístico possui uma resposta categórica, que é geralmente binária, ao invés de uma resposta real.

Para o nosso experimento, iremos utilizar o algoritmo **LogisticRegression** disponível pela biblioteca [scikit-learn](#), com os seguintes parâmetros *default*:

- penalty: 'l2'
- dual: False
- tol: 0.0001
- C: 1.0
- fit\_intercept: True
- intercept\_scaling: 1
- class\_weight: None
- random\_state: None
- solver: 'lbfgs'

- max\_iter: 100
- multi\_class: 'auto'
- verbose: 0
- warm\_start: False
- n\_jobs: None
- l1\_ratio: None

Os resultados que nós obtivemos após o processo de validação 4.2.1 será apresentado na figura 22 abaixo. As acurácias apresentadas são a média de todas acurácias do processo de *cross-validation*.



Figura 22 – Resultados Logistic Regression

Com a finalidade de compreender melhor o desempenho preditivo do nosso modelo, criamos algumas variantes do mesmo, são elas:

- Predição Pré-Jogo
- Predição aos 10 minutos utilizando somente dados da partida
- Predição aos 15 minutos utilizando somente dados da partida

Os resultados que nós obtivemos após o mesmo processo de validação 4.2.1 realizado

anteriormente será apresentado na figura 23 a seguir:

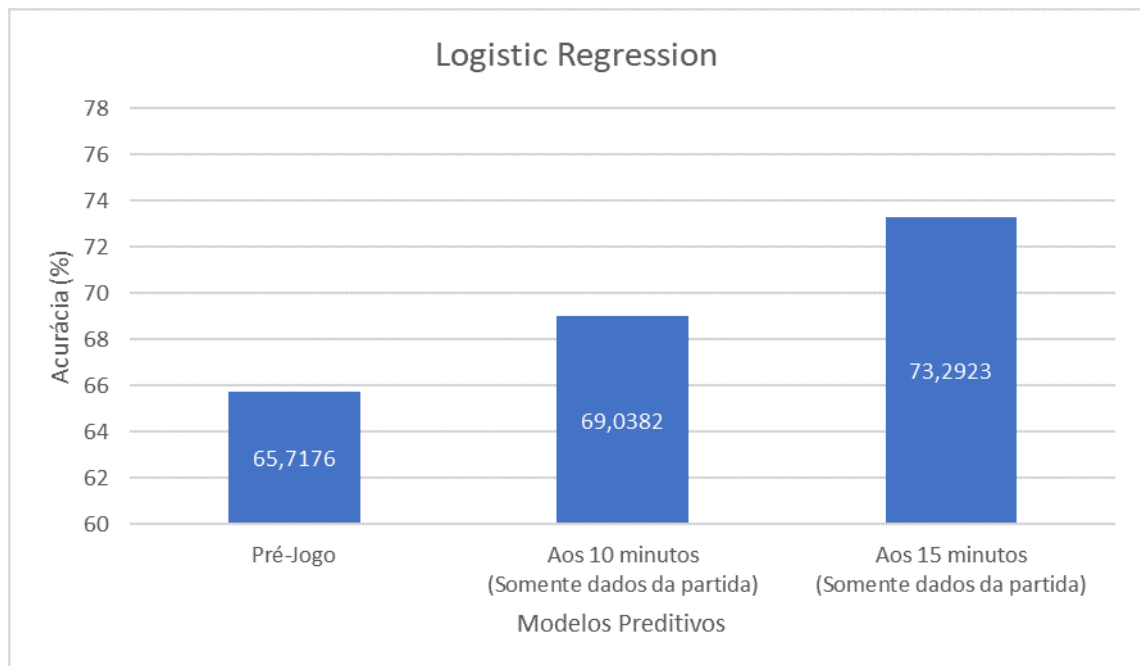


Figura 23 – Resultados Variantes Logistic Regression

#### 4.2.7 Random Forest

*Random Forest*, como o próprio nome diz, é um algoritmo que irá criar árvores de decisão, de maneira aleatória, formando o que podemos enxergar como uma floresta, onde cada árvore será utilizada na escolha do resultado final. As Árvores de Decisão, por sua vez é um algoritmo que criará um fluxograma, com “nós” onde uma condição é verificada, se atendida o fluxo segue por um ramo, caso contrário, por outro, sempre levando ao próximo nó, até a finalização da árvore.

Para o nosso experimento, iremos utilizar o algoritmo **RandomForestClassifier** disponível pela biblioteca [scikit-learn](#), com os seguintes parâmetros *default*:

- `n_estimators`: 100
- `criterion`: 'gini'
- `max_depth`: None
- `min_samples_split`: 2
- `min_samples_leaf`: 1
- `min_weight_fraction_leaf`: 0.0
- `max_features`: 'auto'

- max\_leaf\_nodes: None
- min\_impurity\_decrease: 0.0
- min\_impurity\_split: None
- bootstrap: True
- oob\_score: False
- n\_jobs: None
- random\_state: None
- verbose: 0
- warm\_start: False
- class\_weight: None
- ccp\_alpha: 0.0
- max\_samples: None

Os resultados que nós obtivemos após o processo de validação 4.2.1 será apresentado na figura 24 abaixo. As acurácias apresentadas são a média de todas acurácias do processo de *cross-validation*.

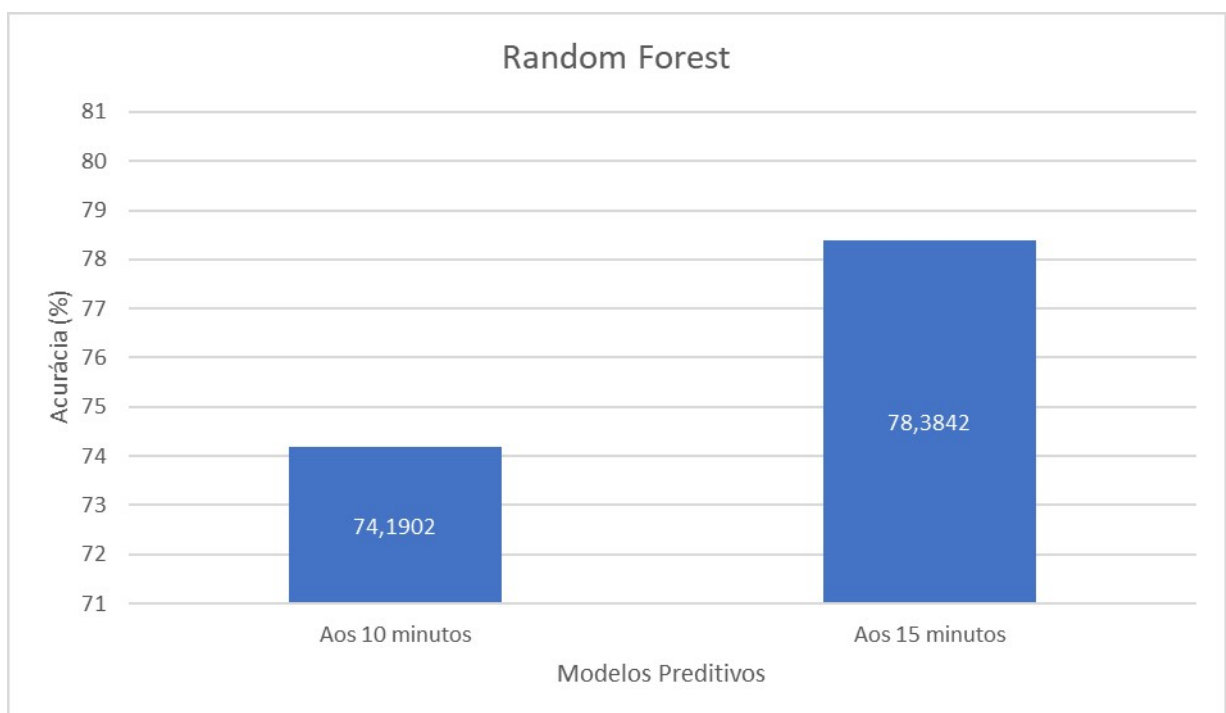


Figura 24 – Resultados Random Forest

Com a finalidade de compreender melhor o desempenho preditivo do nosso modelo, criamos algumas variantes do mesmo, são elas:

- Predição Pré-Jogo
- Predição aos 10 minutos utilizando somente dados da partida
- Predição aos 15 minutos utilizando somente dados da partida

Os resultados que nós obtivemos após o mesmo processo de validação 4.2.1 realizado anteriormente será apresentado na figura 25 a seguir:

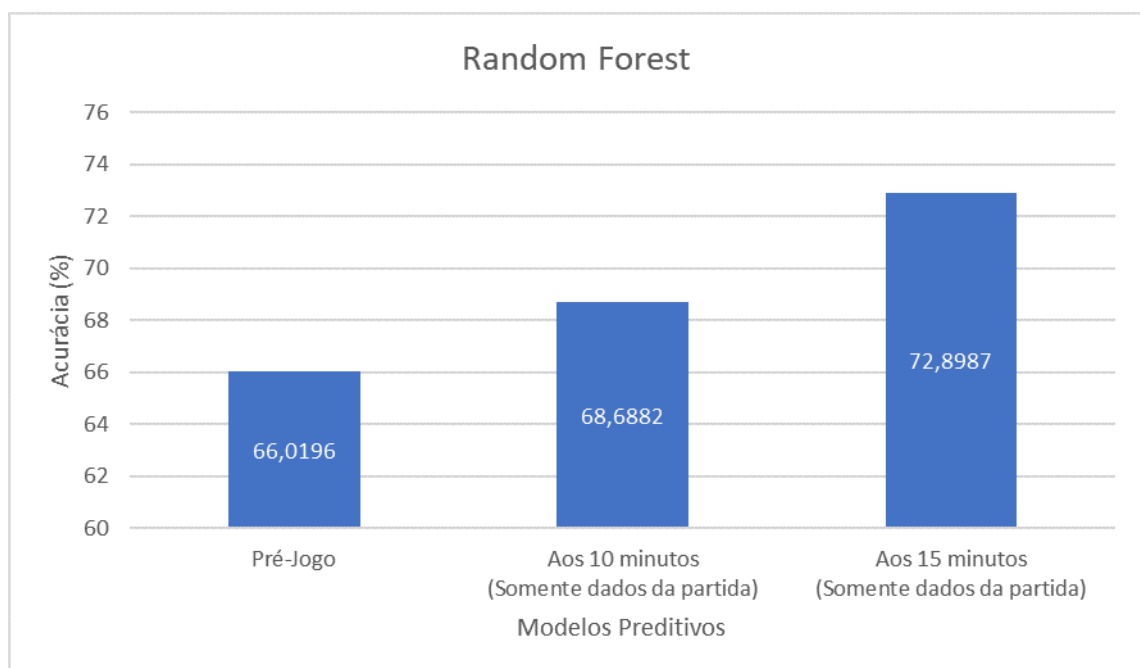


Figura 25 – Resultados Variantes Random Forest

#### 4.2.8 Escolha dos Modelos

Após a realização dos experimentos, podemos comparar os resultados que obtivemos para os diferentes modelos. As acurácias das predições aos 10 minutos serão apresentadas na figura 26 e aos 15 minutos na figura 27.

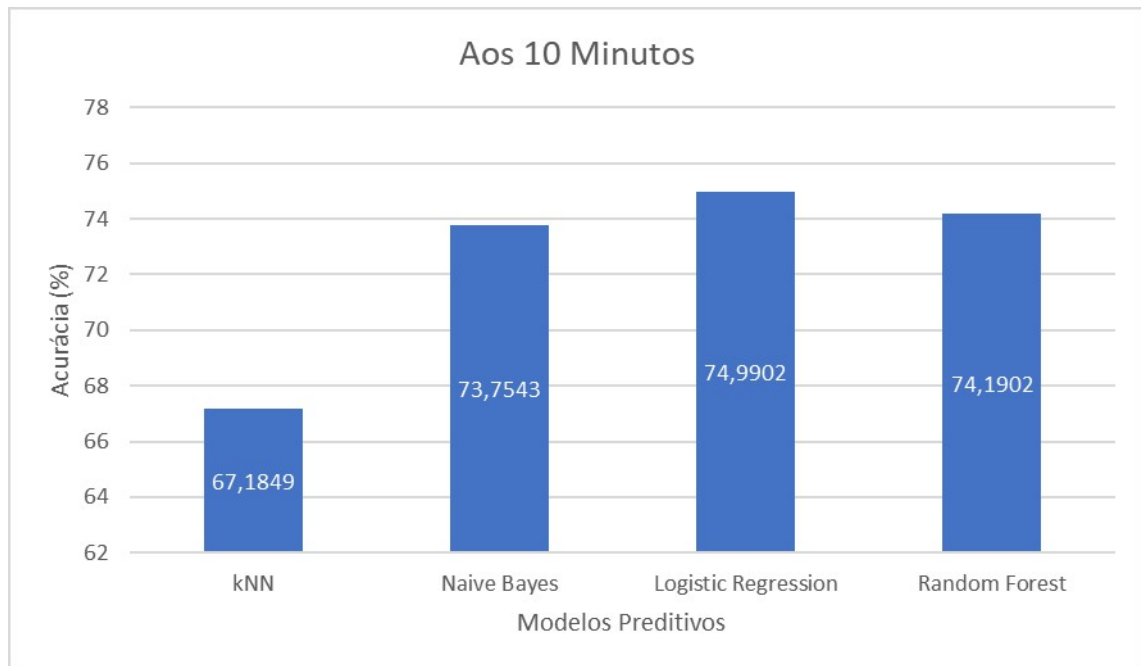


Figura 26 – Resultados Aos 10 Minutos

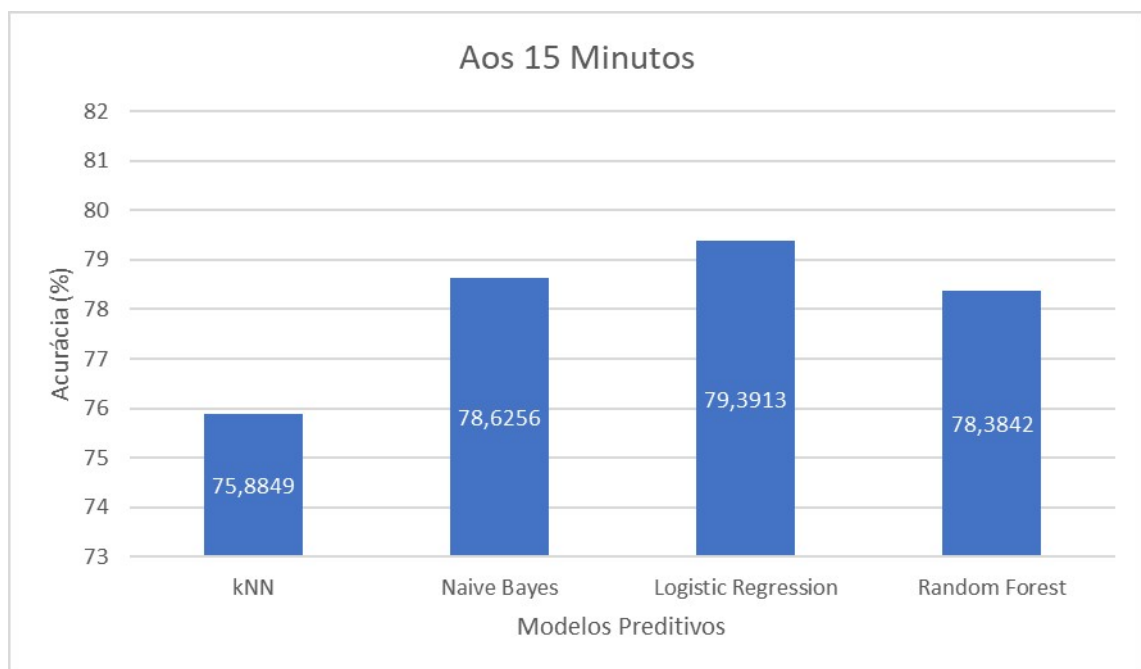


Figura 27 – Resultados Aos 15 Minutos

Como podemos perceber três dos quatro modelos tiveram acurácias parecidas tanto para predição aos 10 minutos, quanto para predição aos 15 minutos, com o modelo que utiliza *Logistic Regression* tendo um acurácia relativamente maior.

Sendo assim decidimos por descartar o modelo kNN, e seguir com os três restantes para fazer um processo de otimização dos mesmos a fim de melhorar seus desempenhos, e assim optar pelo modelo final que será utilizado no nosso trabalho.



## 4.3 Otimização

Modelos de *machine learning* são compostos por parâmetros que fazem parte de sua formulação matemática, onde valores são estimados a partir de um conjunto de dados de treinamento. Nestes modelos, existem ainda parâmetros denominados Hiperparâmetros, que se diferenciam dos primeiros por não serem estimados do mesmo modo e por necessitarem de uma definição de valores antes mesmo que do treinamento.

Os hiperparâmetros definem propriedades a complexidade do modelo e o quão rápido os parâmetros serão aprendidos. Ou seja, os hiperparâmetros estão diretamente ligados ao desempenho do modelo treinado e ao número de operações computacionais necessárias no aprendizado.

Nesta seção será introduzido o método utilizado para otimização hiperparamétrica, assim como a otimização dos modelos selecionados na fase inicial de experimentos 4.2.8, apresentando a definição estrutural destes modelos, assim como seus hiperparâmetros otimizados e seus respectivos incrementos no desempenho de predição.

### 4.3.1 Grid Search

Tradicionalmente, a otimização de hiperparâmetros tem sido uma tarefa humana, pela sua grande eficiência quando apenas poucos hiperparâmetros estão presentes no problema de aprendizado.

Entretanto, a exploração manual do espaço hiperparamétrico, na maior parte dos casos, é uma tarefa que se pode levar desde horas a dias dependendo do conjunto de dados, e está inclinada a resultar em um desempenho insatisfatório do modelo.

Atualmente há um grande apelo pela por métodos automáticos para realizar a otimização hiperparamétrica de modelos. Existem várias abordagens diferentes com este problema, em nosso trabalho foi selecionado o *Grid Search*.

*Grid Search* é uma técnica de ajuste que a partir de uma lista de valores possíveis, tenta calcular os hiperparâmetros ideais realizando uma pesquisa exaustiva. Para fazermos isso iremos utilizar a função **GridSearchCV** disponibilizado pela biblioteca [scikit-learn](#).

### 4.3.2 Naive Bayes

Para otimizar nosso modelo, iremos utilizar a seguinte lista de parâmetros:

- `var_smoothing`: [1, 0.1, 0.001, 0.0001, 0.00001, 0.000001, 0.0000001, 0.00000001, 0.000000001, 0.0000000001]

Assim como nos experimentos, a validação dos modelos otimizados foi feita utilizando

*Repeated k-fold Cross-Validation*, explicado anteriormente na seção 4.2.1. Os resultados que nós obtivemos será apresentado na figura 28 abaixo. As acurácias apresentadas são disponibilizadas pela função *best\_score* do *GridSearchCV*.

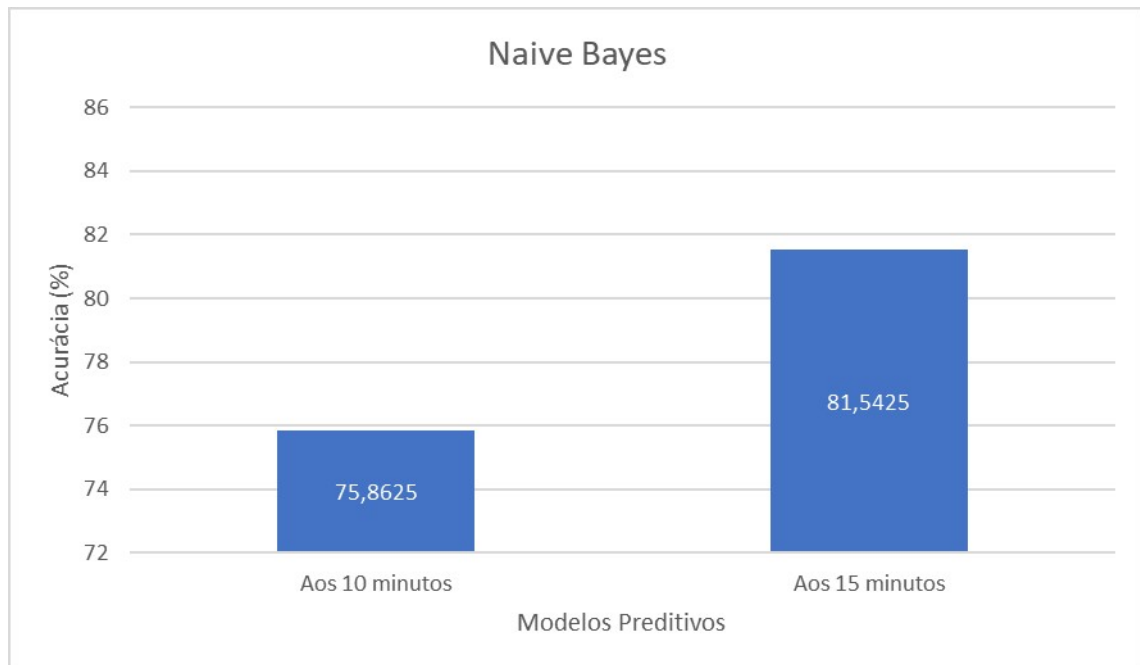


Figura 28 – Resultados Otimizados Naive Bayes

Nosso *Grid Search* identificou utilizando a função *best\_params* do *GridSearchCV* como os melhores parâmetros sendo:

Tabela 7 – Melhores Parâmetros - Naive Bayes

Parâmetro	Predição aos 10 minutos	Predição aos 15 minutos
var_smoothing	0.8154255319148936	0.7586250000000001

### 4.3.3 Logistic Regression

Para otimizar nosso modelo, iremos utilizar a seguinte lista de parâmetros:

- C: [0.001, 0.01, 0.1, 1, 10, 100, 1000]
- penalty: ['l1', 'l2']
- max\_iter: [100, 200, 300, 400, 500, 600, 700]
- solver: ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']

Assim como nos experimentos, a validação dos modelos otimizados foi feita utilizando *Repeated k-fold Cross-Validation*, explicado anteriormente na seção 4.2.1. Os resultados

que nós obtivemos será apresentado na figura 29 abaixo. As acurácias apresentadas são disponibilizadas pela função *best\_score* do *GridSearchCV*.

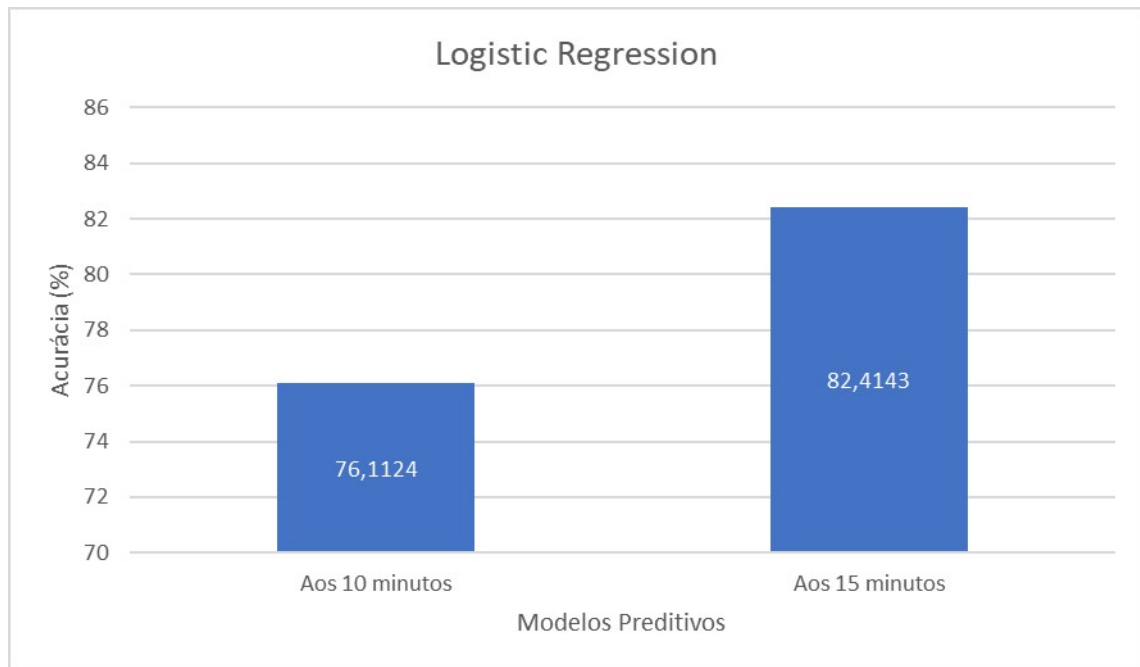


Figura 29 – Resultados Otimizados Logistic Regression

Nosso *Grid Search* identificou utilizando a função *best\_params* do *GridSearchCV* como os melhores parâmetros sendo:

Tabela 8 – Melhores Parâmetros - Logistic Regression

Parâmetro	Predição aos 10 minutos	Predição aos 15 minutos
C	10	0.01
penalty	'l1'	'l2'
max_iter	100	100
solver	'liblinear'	'liblinear'

#### 4.3.4 Random Forest

Para otimizar nosso modelo, iremos utilizar a seguinte lista de parâmetros:

- `bootstrap`: [True, False]
- `criterion`: ['entropy', 'gini']
- `max_depth`: [2, 5, 10, 20, 40, 80, 100, 200, 400, None]
- `max_features`: ['auto', 'sqrt', 'log2', None]

- min\_samples\_leaf: [1, 2, 4, 6, 8]
- min\_samples\_split: [2, 5, 10, 20, 40]
- n\_estimators: [10, 20, 40, 80, 100, 120, 240]

Assim como nos experimentos, a validação dos modelos otimizados foi feita utilizando *Repeated k-fold Cross-Validation*, explicado anteriormente na seção 4.2.1. Os resultados que nós obtivemos será apresentado na figura 30 abaixo. As acurácias apresentadas são disponibilizadas pela função *best\_score* do *GridSearchCV*.

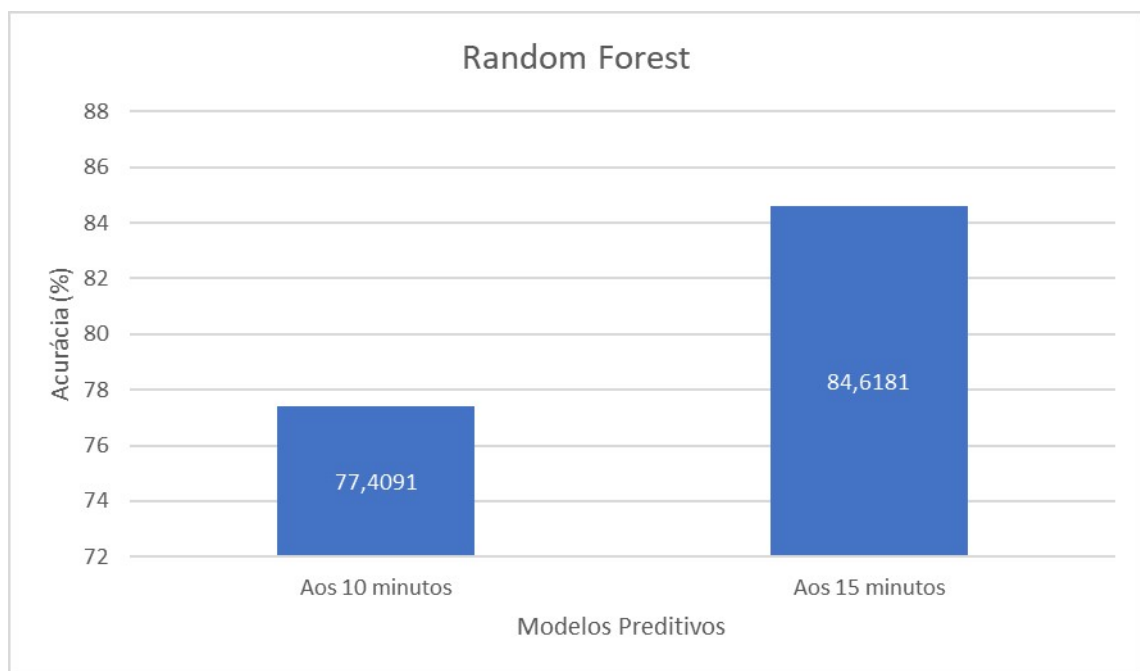


Figura 30 – Resultados Otimizados Random Forest

Nosso *Grid Search* identificou utilizando a função *best\_params* do *GridSearchCV* como os melhores parâmetros sendo:

Tabela 9 – Melhores Parâmetros - Random Forest

Parâmetro	Predição aos 10 minutos	Predição aos 15 minutos
bootstrap	True	True
criterion	'entropy'	'gini'
max_depth	5	10
max_features	'sqrt'	'auto'
min_samples_leaf	1	2
min_samples_split	10	10
n_estimators	240	20

### 4.3.5 Modelo Final

Após a realização das otimizações, podemos comparar os resultados que obtivemos para os diferentes modelos. As acurácias das predições aos 10 minutos serão apresentadas na figura 31 e aos 15 minutos na figura 32.

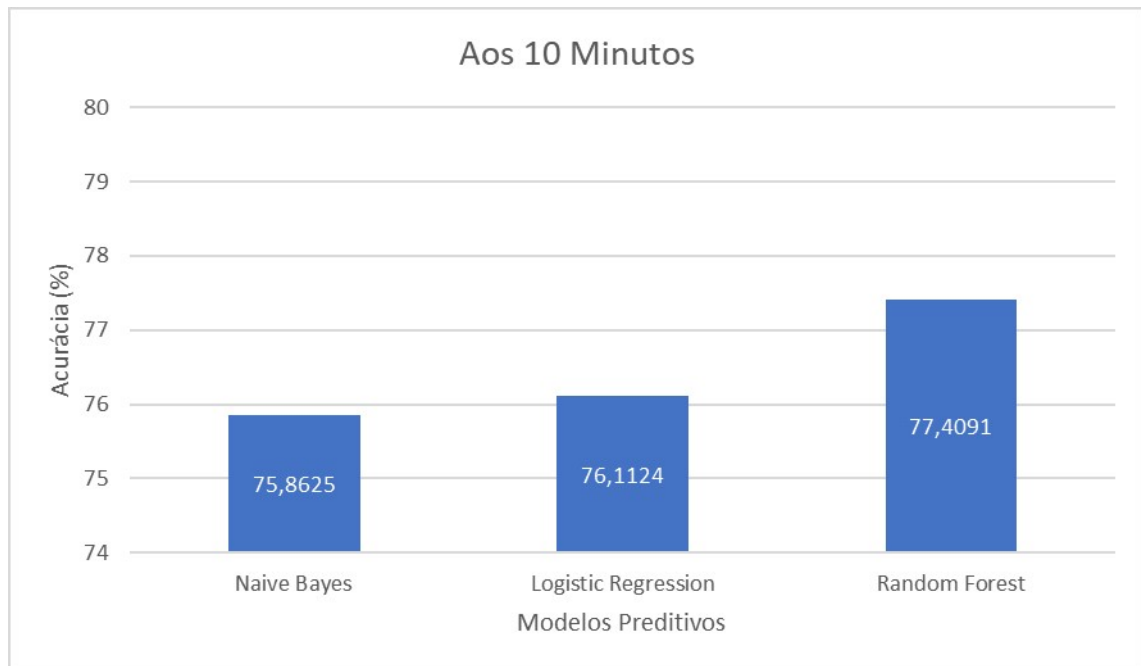


Figura 31 – Resultados Otimizados Aos 10 Minutos

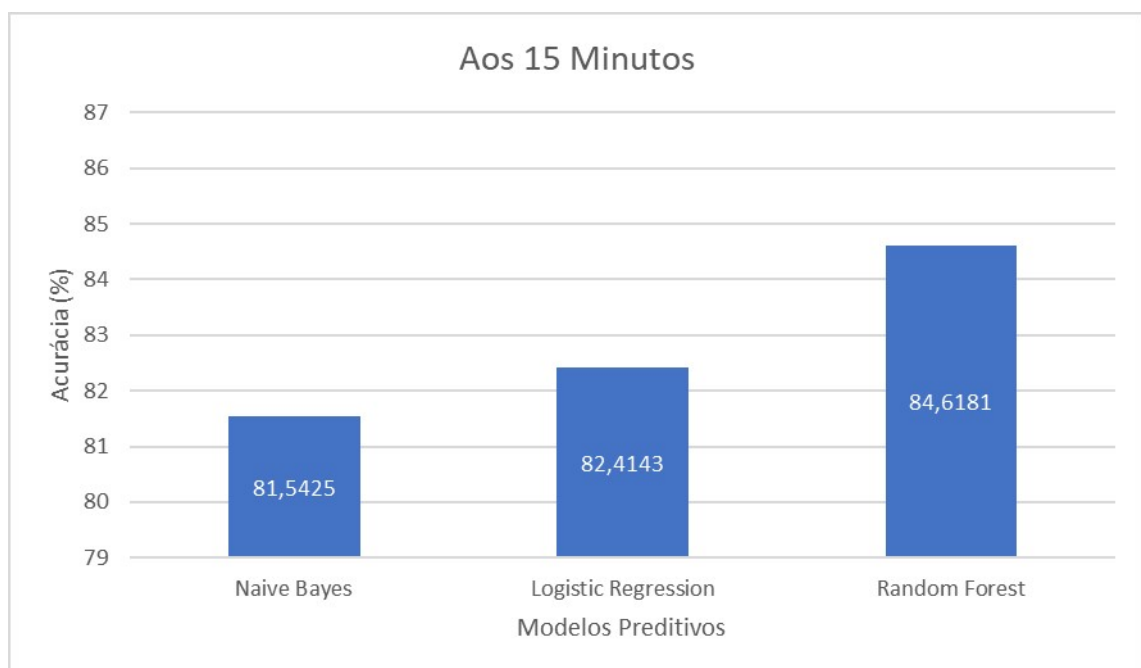


Figura 32 – Resultados Otimizados Aos 15 Minutos

Como podemos perceber, tanto para predição aos 10 minutos, quanto para predição

aos 15 minutos, os três modelos tiveram uma melhora relativamente grande de acurácia comparada aos anteriores 4.2.8, com o modelo que utiliza *Random Forest*, apesar de não ter sido o com melhor desempenho anteriormente, tendo um acurácia relativamente maior em ambas predições.

Sendo assim, definimos o *Random Forest* como nosso modelo final, com predição aos 10 minutos de 77,4091% de acurácia e predição aos 15 minutos de 84,6181% de acurácia.

#### 4.3.6 Comparações Gerais

Para podermos compreender melhor o desempenho dos nossos modelos finais, iremos compará-los com suas versões anteriores, na qual tiveram as melhores acurácias, são elas:

- Predição Pré-Jogo (*Random Forest*)
- Predição aos 10 minutos utilizando somente dados da partida (*Logistic Regression*)
- Predição aos 15 minutos utilizando somente dados da partida (*Logistic Regression*)
- Predição aos 10 minutos não otimizado (*Logistic Regression*)
- Predição aos 15 minutos não otimizado (*Logistic Regression*)

A figura 33 a seguir apresenta os resultados:

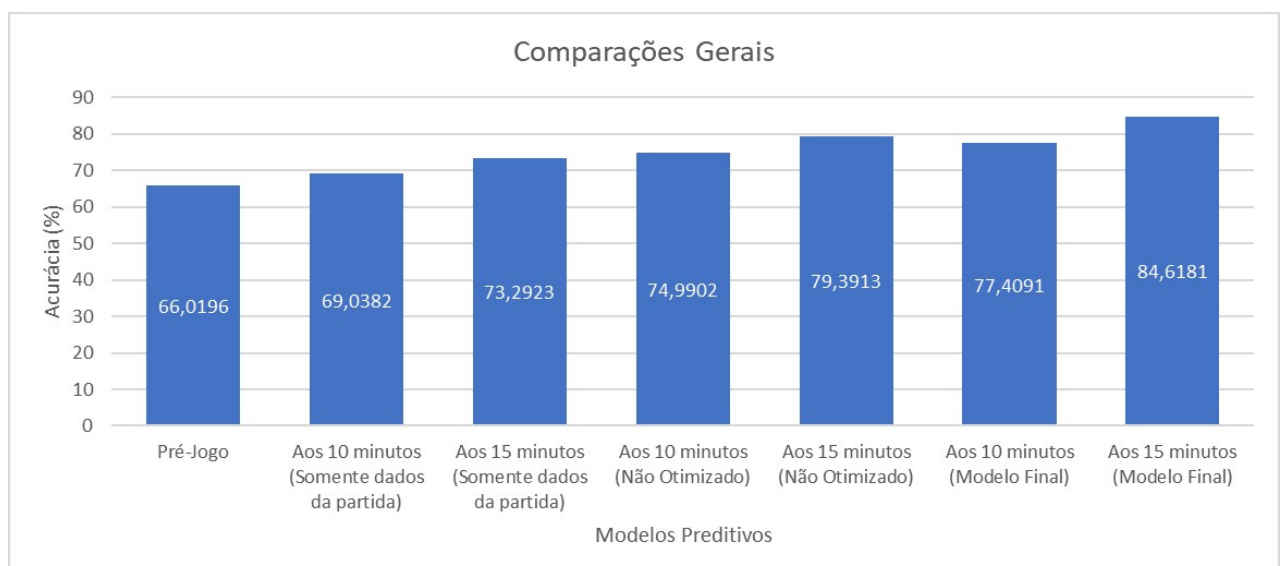


Figura 33 – Comparações Gerais

## 4.4 Avaliação de Lucro nas Apostas

Como estabelecemos na subseção 2.1.3 uma estratégia lucrativa para conseguir gerar lucro no longo prazo das apostas, nosso trabalho é encontrar vantagens em relação as probabilidades oferecidas pelas casas de apostas e as probabilidades reais para um evento.

No nosso caso, utilizando o modelo final, temos que para uma determinada aposta em um resultado da partida a probabilidade de ela estar certa é de 77,4091% aos 10 minutos e 84,6181% aos 15 minutos. Para convertermos para o equivalente em *odds* temos que dividir 1 pela porcentagem dividida por 100. Por exemplo, uma probabilidade de 50% =  $1 / (50 / 100) = 2$ . Ou seja, para nosso modelo, temos uma *odd* de 1,29 aos 10 minutos e uma *odd* de 1,18 aos 15 minutos.

Uma possível estratégia para gerar lucro, seria o investidor utilizar o seguinte método, apostar somente aos 10 minutos utilizando a predição do nosso modelo. Para que esta estratégia faça sentido, teríamos que encontrar apostas, onde as casas estão estipulando que o vencedor da partida tem menos chance de vencer do que 77,4091%, ou seja, uma *odd* de pelo menos 1.30.

Tendo isso em mente, suponhamos então um cenário onde o investidor realizaria 1000 apostas seguindo esta estratégia, onde seria apostado R\$100,00 em determinados times que estariam com uma *odd* de 1.50. Então, ao final destas apostas, seguindo a probabilidade estatística da predição estar certa, o investidor acertaria 770 apostas. Temos então que cada aposta vencedora retornaria um lucro de R\$50,00 pela *odd* 1.50, ou seja, no total seriam  $770 * 50 = \text{R\$}38.500,00$  de ganhos com as apostas corretas, porém teriam  $230 * 100 = \text{R\$}23.000,00$  de perdas com as apostas incorretas. O saldo final seria os ganhos com as apostas corretas - as perdas com as apostas incorretas =  $\text{R\$}18.500,00$ .

Vimos então que, caso sejam sempre feitas apostas utilizando um método eficaz, encontrando valor nas apostas, por mais que não acerte todos os resultados e até podendo ter prejuízo em uma pequena quantidade inicial de partidas, ao longo prazo terá lucro, já que como suas apostas tem mais chance de acontecer do que a estipulada pela casa, o lucro potencial em cima dessas apostas é maior do que deveria.

## 5 Conclusão

Neste capítulo serão apresentadas as principais contribuições do trabalho, assim como os desafios e soluções encontradas, as lições aprendidas durante o desenvolvimento e as perspectivas de trabalhos futuros.

### 5.1 Considerações Finais

Este trabalho tinha como proposta principal desenvolver um modelo de classificação capaz de realizar predições e entregar ao usuário esse resultado. Utilizamos diversas técnicas de *machine learning*, como *k-Nearest-Neighbors*, *Naive Bayes*, *Logistic Regression* e *Random Forest Classifier* para gerar o resultado de uma partida.

Tivemos que realizar um levantamento de material bibliográfico, procurando por artigos, blogs especializados, livros, fóruns, com o intuito de estudar o mercado de apostas, desenvolvimento web, técnicas de inteligência artificial, algoritmos de *machine learning* e análise de dados.

Conseguimos encontrar conjuntos de dados contendo as informações necessárias, prepara-los e otimiza-los para o nosso trabalho.

Criamos um método de teste, com o intuito de obter o melhor desempenho possível, e então realizamos experimentos para comparar os resultados das diversas técnicas, e encontrar as que mais se adequam ao nosso problema.

Realizamos otimizações em nossos modelos utilizando um método de otimização hiperparamétrica, e então encontramos o nosso modelo final com o melhor desempenho possível.

Comparamos nosso resultado final com os modelos anteriores, a fim de compreender melhor o desempenho preditivo. Descobrimos que nosso modelo final apresenta acurácia satisfatoriamente melhor do que sua versão não otimizada, assim como suas versões utilizando somente dados em tempo real, e utilizando somente dados pré-jogo.

Então criamos um *Back-End* para ficar responsável por preparar os dados e entregá-los como entrada para o nosso Modelo, que retornará uma predição para ser exibida para o usuário.

Construímos uma aplicação web que possibilite ao usuário visualizar informações relevantes para realizar as suas apostas, desde os históricos dos times, até as estatísticas da partida, assim como a predição em tempo real.

Além disso, instruímos ao investidor uma estratégia lucrativa de aposta, de modo que



ao utilizar o nosso modelo de predição conseguimos encontrar valor em uma aposta que teve sua probabilidade estipulada pela casa de aposta.

Um dos principais desafios enfrentados durante o trabalho foi encontrar um conjunto de dados adequados para construir um modelo satisfatório. Muito tempo foi gasto fazendo pesquisas para encontrar dados disponibilizados gratuitamente e que condissesse com o nosso problema.

Uma outra dificuldade encontrada neste estudo foi a realização dos experimentos com uma limitada quantidade de recursos computacionais.

## 5.2 Trabalhos Futuros

Embora alcançado nosso objetivo principal, existem oportunidades para melhorar nosso modelo e alguns outros caminhos no qual esse projeto pode ser levado com mais tempo e recursos. A seguir serão listadas sugestões para trabalhos futuros relacionados ao tema deste estudo:

### 5.2.1 Melhores Conjuntos de Dados

Primeiramente, acreditamos que o modelo definitivamente pode ser melhorado com dados que possuam mais variáveis correlatas ao resultado final do jogo. Por exemplo, alguns serviços provedores de dados como o [PandaScore](#) disponibilizam dados sobre cada jogador dos times, onde é possível estimar o seu *overall*, a sua proficiência com determinado Campeão que irá disputar a partida. Tendo acesso a essas informações claramente haveria um aumento no desempenho do modelo.

Também não temos acesso em relação aos Campeões que irão disputar uma partida, caso saibamos qual a porcentagem de vitória que os Campeões possuem no momento em que a partida foi disputada, poderíamos saber antes de a partida começar de fato, qual time iniciaria com a vantagem, visto que times com melhores Campeões possuem maior chance de vencer do que seus adversários.

### 5.2.2 Modelos Preditivos Para Outros Mercados

No mercado de apostas esportivas existem além do resultado final da partida, diversos outros mercados, como por exemplo, quantidade total ao final da partida de torres, barões, dragões ou eliminações, além de mercados pré-jogo, como por exemplo, quem fará o *first blood*, ou quem levará a primeira torre. Todas essas predições são possíveis de serem feitas construindo modelos de classificação utilizando esse mesmo conjunto de dados.

### 5.2.3 Validação Lucrativa das Apostas

Neste trabalho, tínhamos como propostas desenvolver um modelo de classificação capaz de realizar previsões, entregar ao usuário esse resultado e instruí-lo com um método de aposta que seja capaz de gerar lucro ao longo prazo, todos esses objetivos foram concluídos, entretanto, a validação do método não foi realizada.

Para realizarmos essa tarefa, precisaríamos popular uma base de dados manualmente. Utilizaríamos o nosso sistema para visualizar as previsões das partidas, e a partir delas, analisaríamos, se essa determinada partida estaria de acordo com o nosso método, caso esteja, ao final da mesma, seria armazenado o possível lucro ou prejuízo que teríamos. Então ao final, quando possuíssemos uma amostra consideravelmente grande de aposta, poderíamos validar a lucratividade do método.

### 5.2.4 Sistema de Automação de Apostas

Outra potencial extensão futura deste projeto poderia ser a automatização das apostas por meio de um sistema. Para tal, precisaríamos ter acesso as *odds* disponibilizadas pelas casas de apostas por meio de provedores, ou através de uma extração de dados utilizando *Web Scraping*, entretanto, a segunda opção teria de ser feita em casas de apostas onde esse tipo de processo é autorizado pela mesma.

Com acesso a essas *odds*, o sistema poderia avaliar se determinada previsão se encaixaria no método devidamente avaliado, então o mesmo ficaria responsável por realizar essa aposta automaticamente, porém, como dissemos anteriormente, essa automação teria de ser feita somente em casas de apostas que possuíssemos autorização.

Poderíamos utilizar para implementar esse sistema de automação de apostas o *framework* Selenium, nele seria possível tanto extrair as *odds*, quanto interagir com o *browser* para realizar uma aposta.

# Referências

Yogesh, Singh & Bhatia, Pradeep & Sangwan, Om. (2007). A REVIEW OF STUDIES ON MACHINE LEARNING TECHNIQUES. International Journal of Computer Science and Security. 1.

John Paul Mueller and Luca Massaron. 2016. Machine Learning For Dummies (1st. ed.). For Dummies.

S. B. Kotsiantis. 2007. Supervised Machine Learning: A Review of Classification Techniques. In Proceedings of the 2007 conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies. IOS Press, NLD, 3–24.

Bunker, Rory & Thabtah, Fadi. (2017). A Machine Learning Framework for Sport Result Prediction. Applied Computing and Informatics. 15. 10.1016/j.aci.2017.09.005.

Moroney M J (1956). Facts from figures. 3rd edition, Penguin, London.

Dixon M J and Coles S G (1997). Modelling association football scores and inefficiencies in the football betting market. Applied Statistics, 46, 2, 265-280

Reep C and Benjamin B (1968). Skill and chance in association football. Journal of the Royal Statistical Society A, 131, 581-585.

Maher M J (1982). Modelling association football scores. Statistica Neerlandica, 36, 109-118.

Ganesan, Anand & Murugan, Harini. (2020). ENGLISH FOOTBALL PREDICTION USING MACHINE LEARNING CLASSIFIERS. International Journal of Pure and Applied Mathematics. 118. 533-536.

Sommerville, I. Engenharia de Software. 9. ed. São Paulo: Pearson Prentice Hall, 2010. Tradução Ivan Bosnic e Kalinga G. de O. Gonçalves.

J. Bosch, “Design and Use of Software Architectures Adopting and Evolving a Product-Line Approach”, New York, 2000, Addison-Wesley.

Kaunitz, Lisandro & Zhong, Shenjun & Kreiner, Javier. (2017). Beating the bookies with their own numbers - and how the online sports betting market is rigged.

Langseth, Helge. (2013). Beating the bookie: A look at statistical models for prediction of football matches. Frontiers in Artificial Intelligence and Applications. 257. 165-174. 10.3233/978-1-61499-330-8-165.

Kain, Kyle & Logan, Trevon. (2012). Are Sports Betting Markets Prediction Markets?: Evi-

dence From a New Test. *Journal of Sports Economics*. 15. 45-63. 10.1177/1527002512437744.

Roxborough, R. & Rhoden, M.. (2021). *Sports Book Management: A Guide for the Legal Bookmaker*.

Breiman, L. Random Forests. *Machine Learning* 45, 5–32 (2001). <https://doi.org/10.1023/A:1010933>

Webb G.I. (2011) Naïve Bayes. In: Sammut C., Webb G.I. (eds) *Encyclopedia of Machine Learning*. Springer, Boston, MA. [https://doi.org/10.1007/978-0-387-30164-8\\_576](https://doi.org/10.1007/978-0-387-30164-8_576)

Zhang, Zhongheng. (2016). Introduction to machine learning: k-nearest neighbors. *Annals of translational medicine* vol. 4,11 : 218. doi:10.21037/atm.2016.03.37