HÉULLER SOARES VILELA SILVA

PREVISÃO DO RESULTADO FINAL DE PARTIDAS DE FUTEBOL A PARTIR DO PRIMEIRO TEMPO POR MEIO DE TÉCNICAS DE APRENDIZAGEM DE MÁQUINA

LAVRAS – MG

2021

HÉULLER SOARES VILELA SILVA

PREVISÃO DO RESULTADO FINAL DE PARTIDAS DE FUTEBOL A PARTIR DO PRIMEIRO TEMPO POR MEIO DE TÉCNICAS DE APRENDIZAGEM DE MÁQUINA

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Lavras, como parte das exigências do curso de Ciência da Computação, para a obtenção do título de Bacharel.

Prof. Dr. Eric Fernandes de Mello Araújo

Orientador

LAVRAS – MG

2021

HÉULLER SOARES VILELA SILVA

PREVISÃO DO RESULTADO FINAL DE PARTIDAS DE FUTEBOL A PARTIR DO PRIMEIRO TEMPO POR MEIO DE TÉCNICAS DE APRENDIZAGEM DE MÁQUINA

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Lavras, como parte das exigências do curso de Ciência da Computação, para a obtenção do título de Bacharel.

APROVADA em 22 de Abril de 2021.

Prof. Dr. Erick Galani Maziero	UFLA
Prof. Dr. Rafael Serapilha Durelli	UFLA

Prof. Dr. Eric Fernandes de Mello Araújo
Orientador

LAVRAS – MG
2021

À minha família por toda dedicação, cuidado e amor. Com toda admiração e gratidão, dedico.

AGRADECIMENTOS

A Deus pelo dom da vida, por me proporcionar momentos que tive grande aprendizado e guardarei para sempre em minha memória.

Aos meus pais Hélcio e Luciene, pelos ensinamentos que formaram meu caráter e também por todo o suporte que nunca me foi negado para realização de desafios em minha vida.

Ao meu irmão Hélcio Jr. que sempre torceu por mim e realiza comigo muito bem todos os dias o papel de irmão mais velho.

À Universidade Federal de Lavras e ao departamento de Ciência da Computação, pela oportunidade da realização de um sonho.

Ao meu orientador, professor Eric, por me apresentar à área de ciência de dados e me orientar com todo apoio, confiança, dedicação e paciência.

Aos meus professores pela competência com que ministraram as respectivas disciplinas e que me ajudaram no processo de desenvolvimento de conhecimento.

Aos colegas de classe que fizeram parte desta caminhada.

Aos verdadeiros amigos que estiveram comigo ao longo destes últimos anos e que, por sua vez, me dispensaram ajuda, compreensão, atenção, paciência e com quem aprendi muito, com certeza ficarão na minha mente para sempre registrados.

Os computadores são capazes de ver, ouvir e aprender. Bem vindo ao futuro.
(Dave Waters)

RESUMO

O futebol é o esporte mais popular do Brasil, e parte importante da cultura do país, atraindo altos investimentos por parte de empresas patrocinadoras e torcedores. A complexidade dos jogos de futebol, onde eventos imprevisíveis como cartões e lesões ocorrem constantemente, gera um grande interesse por parte de casas de apostas e trabalhadores envolvidos com o esporte, como jornalistas e comissões técnicas. Esse interesse é majoritariamente na tentativa de prever resultados, visando ganhar apostas ou melhor preparar uma equipe para enfrentar seu adversário. As informações prévias do jogo, como o histórico entre os times, os resultados mais recentes e a escalação inicial certamente são fatores que podem ser usados na tentativa de prever o resultado final da partida. Com a popularização de técnicas de aprendizagem de máquina, vários trabalhos vêm sendo desenvolvidos na tentativa de fazer essa previsão considerando o máximo de informações possível. Este trabalho estudou três modelos distintos de predição de resultados de partidas de futebol a partir das estatísticas do primeiro tempo, como Regressão Logística, Gradient Boosting e Floresta Aleatória. Os resultados mostraram que o algoritmo Regressão Logística obteve o melhor resultado com 57% de acurácia, um desempenho relativamente parecido com outros trabalhos que visam fazer a mesma análise. Esse trabalho também desenvolveu uma ferramenta para, em tempo real, mostrar as probabilidades de vitória, empate ou derrota da equipe que joga em casa usando o modelo de melhor desempenho.

Palavras-chave: aprendizagem de máquina. inteligência artificial. futebol.

ABSTRACT

Football is the most popular sport in Brazil and an important part of its culture. Therefore, it attracts a huge amount of investment from sponsoring companies and fans. The events in football matches, such as yellow/red cards and injuries, make it complex and unpredictable. As a result, bookmarkers and the football community are deeply interested in these data. This interest is mainly in the attempt to predict results to win bets or prepare a team to beat their rival. The previous match results, statistics, and the initial line-up are certain factors that can be used to predict the final result of a game. With the popularization of machine learning techniques, several studies have been developed to improve football prediction, considering as much information as possible. This work studied three different models for predicting the results of football matches based on the statistics of the first half, such as Logistic Regression, Gradient Boosting, and Random Forest. The results showed that the Logistic Regression algorithm obtained the best result with 57% accuracy, a performance relatively similar to other studies that aim to make the same analysis. This work also developed a tool to show, in real-time, the chances of victory, draw, or defeat of home teams using the best performance model.

Keywords: machine learning. artificial inteligence. football.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Futebol é o esporte mais assistido no Brasil, sendo parte central da cultura brasileira e grande foco de investimentos financeiros por parte de empresas. O Palmeiras, um dos times mais tradicionais de futebol do Brasil, apresenta um contrato milionário com a empresa Crefisa, que segundo GOAL (2021) prevê pagamento de 81 milhões de reais anuais pela exclusividade da marca na camisa, além de bonificações por metas que poderiam chegar a 34 milhões de reais mensais .

De acordo com ONLINE (2020) as casas de apostas movimentam uma grande quantidade de dinheiro estimado em cerca de 4 bilhões de reais por ano no Brasil. Por ser um jogo complexo, onde vários eventos, como lesões ou expulsões, podem alterar diretamente a dinâmica da partida, muito se questiona sobre as possibilidades de prever o resultado final a partir de informações prévias.

O crescimento no uso de aprendizagem de máquina para a resolução de problemas em vários campos também encontra espaço nos esportes. (AZEVEDO, 2019; NABINGER, 2018; GONZÁLEZ, 2014)

Apesar da dificuldade de acesso aos dados referentes a partidas de futebol, é possível fazer a coleta usando sites que fornecem dados de jogos já realizados e transmitem partidas em tempo real, bem como probabilidades de resultados baseados em diferentes métricas.

Este trabalho se propõe a usar os algoritmos *Gradient Boosting*, Regressão Logística e Floresta Aleatória para prever resultados de uma partida de futebol a partir de informações do primeiro tempo, ou seja, durante o intervalo.

O objetivo geral do trabalho é prever o resultado final de uma partida a partir das estatísticas do primeiro tempo com as melhores métricas.

Os objetivos específicos são coletar uma gama de dados para criar um banco de dados capaz de fornecer dados para o treinamento do modelo e identificar quais variáveis possuem maior importância no momento da previsão. Os resultados obtidos neste trabalho mostraram que é possível prever derrota, empate

e vitória com uma acurácia de 57.25%, precisão de 68%, 44% e 64% respectivamente, *recall* de 60%, 53% e 59% e por fim, f1-score de 63%, 48% e 61%. Os resultados se mostram satisfatórios quando comparados com outros trabalhos desenvolvidos anteriormente.

O Capítulo 2 apresenta ao leitor o referencial teórico para compreensão do trabalho desenvolvido. Também neste capítulo serão apresentados os trabalhos relacionados ao proposto neste documento. O Capítulo 3 aponta os métodos usados para a coleta, tratamento, seleção e predição de resultados das partidas a partir da base de dados utilizada. O Capítulo 4 apresenta os resultados obtidos por meio da aplicação dos algoritmos de aprendizagem de máquina, e o Capítulo 5 discute as implicações dos resultados obtidos, bem como apresenta os potenciais trabalhos futuros a partir deste trabalho.

2 EMBASAMENTO TEÓRICO

Neste capítulo, são apresentados os principais conceitos, técnicas e informações necessárias para o melhor entendimento de como o trabalho foi desenvolvido, e de como interpretar os resultados obtidos. Após apresentar os conceitos, será mostrado como o nosso trabalho se localiza no corpo de estudos feitos usando técnicas de aprendizagem de máquina para fins de predição de resultados esportivos.

2.1 Aprendizagem de máquina

Segundo MONARD; BARANAUSKAS (2003a), o termo aprendizagem de máquina refere-se à área de inteligência artificial na qual se desenvolvem técnicas computacionais sobre o aprendizado, de modo que seja possível criar sistemas que adquirem conhecimento automaticamente. É possível entender o aprendizado de máquina como um sistema que toma decisões com base em experiências acumuladas. Os sistemas de aprendizado de máquina possuem características diferentes de acordo com seu funcionamento. Os tipos de algoritmos de aprendizagem de máquina podem ser (1) supervisionado, (2) não supervisionado e (3) semi-supervisionado e (4) por reforço.

O aprendizado supervisionado (1), de acordo com MONARD; BARANAUSKAS (2003a), consiste em construir um classificador por meio de um conjunto de exemplos rotulados. Dessa forma, novas entradas serão preditas com o intuito de serem classificadas corretamente a partir desse classificador. Diante disso, o classificador será avaliado conforme sua performance nessa tarefa. Normalmente, os registros rotulados possuem características que os definem, além de uma classe que descreve o fenômeno de interesse. Dessa forma, há dois tipos de atributos: nominais, em que não há uma ordenação entre os valores (por exemplo: cor, pode ser amarelo, vermelho, preto) e há contínuos, para os quais uma ordem linear nos valores (por exemplo: peso, 70.5kg, 76.0kg). O algoritmo a partir dos

dados de treinamento produz uma função inferida, em que é usada para mapeamento de novos exemplos. O cenário ideal permitiria ao algoritmo acertar a classe de registros que não estavam na base para treinamento. Um exemplo, segundo BUDKEWICZ (2018) para o aprendizado supervisionado seria testar se e-mails são spam ou não a partir do treinamento com registros rotulados.

O aprendizado não supervisionado (2) diferencia-se do aprendizado supervisionado por não possuir rotulações (uma classe) para cada registro da base de dados. Dessa forma, o algoritmo analisa se é possível formar agrupamentos ou *clusters* baseado nas características de cada registro. Segundo SANCHES (2003), a partir dos agrupamentos realizados é preciso identificar e determinar o que cada agrupamento significa no contexto do problema. Um exemplo, segundo BUDKEWICZ (2018) desse tipo de modelo é a identificação de grupos de clientes de um e-commerce para montar e-mail marketing e vitrines personalizadas.

O aprendizado semi-supervisionado (3) possui propriedades de ambos os modelos apresentados anteriormente. Essa abordagem, assume que existem dois conjuntos de registros, rotulados e não rotulados no momento do treinamento, a qual propõe-se criar um classificador a partir de uma gama de registros não rotulados ao lado de um conjunto rotulado menor. Existem duas formas de ação quando há a presença desses conjuntos: a classificação ou *clustering* semi-supervisionado. Na classificação a ideia antes do treinamento dos dados é rotular com uma certa margem de segurança os dados não rotulados, e no *clustering* os exemplos rotulados são utilizados no processo de formação dos *clusters*. Hoje há a existência de diversas propostas de algoritmos que se baseiam em algum algoritmo existente na literatura. Por exemplo, testar se e-mails são spam ou não a partir de dois conjuntos, sendo um rotulado e o outro não.

Por fim, o aprendizado por reforço (4), de acordo com SUTTON; BARTO (2018) a aprendizagem por reforço é uma técnica de Inteligência Artificial que viabiliza que um agente aprenda a partir da sua interação com o ambiente a qual

está inserido. A aprendizagem se dá por meio do conhecimento sobre o estado do indivíduo no ambiente e das mudanças de estado provenientes de suas ações. Portanto, segundo HONDA; FACURE; YAOHAO (2017) a aprendizagem por reforço utiliza uma estrutura composta de estados, ações e recompensas.Por exemplo, a adestração de um cachorro. Inicialmente, dificilmente o cachorro irá atender sua ordem. Dessa forma, é preciso dar uma "punição", repreendo-o verbalmente. Em situações futuras, a qual o cão agir perto do esperado, você dará uma recompensa, de modo que ele entenda que essa é a forma ideal de comportamento.

Os trabalhos que se utilizam de algoritmos de aprendizado de máquina exploram a construção de sistemas que podem aprender e fazer previsões a partir dos dados fornecidos. De acordo com RON; FOSTER (1998), os algoritmos possuem uma construção a partir de exemplos de entradas para realizar suas previsões. O aprendizado de máquina possui uma forte relação com métodos estatísticos também usadas para fazer previsões. Além disso, sistemas de aprendizagem de máquina podem ser utilizados em situações nas quais projetar e programar é inviável devido à falta de conhecimento sobre as regras que determinam o contexto estudado ou em cenários em que não se consegue pensar em todas regras.

2.1.1 Coleta de dados

A extração de dados pode ser feita de diversas formas, sendo as mais conhecidas as coletas via APIs (sistema de fácil integração entre sistemas que nesse caso dispõe a oferecer um serviço que retorna dados), banco de dados e *crawler* (bot responsável por extrair dados de uma página web). APIs oferecidas por empresas como Twitter, Youtube e Instagram são a maneira mais prática de coleta de dados, pois os dados estarão em um formato acessível para a coleta. A desvantagem está com relação a limitação de atributos e da quantidade de registros oferecidos. Também é possível, utilizando-se do banco de dados da empresa contratante, de forma que a extração será feita por comandos em um banco não relacional ou

comandos SQL em um banco relacional. Por fim a coleta utilizando *crawler*, que foi escolhida para este trabalho, em que é a coleta automatizada em sites, que também pode ser chamada de raspagem de dados. A raspagem de dados de acordo com Moraes (2018) consiste na extração de dados relevantes de um determinado site para uma análise posterior. Há possibilidade desse processo ser feito manualmente, entretanto, devido ao grande trabalho requerido, é mais viável que a coleta seja automatizada.

Segundo MITCHELL (2018), embora os navegadores sejam úteis para exibir imagens e organizar objetos em um formato mais legível, a raspagem de dados possui ótimas ferramentas para processar grande quantidade de dados. Em vez de visualizar uma página por vez, as ferramentas permitem a visualização de várias páginas automaticamente. Entretanto, alguns sites a fim de proteger seus dados podem dificultar esse processo.

2.1.2 Tratamento e Transformação dos dados (pré-processamento)

A fase de pré-processamento inicia-se após os dados coletados e organizados na forma de um conjunto de dados. De acordo com BATISTA et al. (2003), há diversos objetivos nessa fase, tais como identificar e tratar dados corrompidos, atributos irrelevantes e valores desconhecidos. Pode haver também o interesse em aprender mais sobre os dados, o que é possível por meio de visualizações gráficas. Essas manipulações realizadas na fase de pré-processamento têm como objetivo preparar os dados para que a fase posterior (extração de conhecimento), seja mais efetiva.

O autor pontua também que, no aprendizado de máquina, a qualidade dos dados é essencial, pois a qualidade do conhecimento extraído deriva-se pela qualidade dos dados na entrada.

2.1.2.1 Tratamento de dados faltantes

BATISTA et al. (2003) relata que um problema normalmente detectado é a presença de dados faltantes, que consiste em atributos que não foram medidos para alguns casos. Esses valores ausentes podem ter diversas fontes, como defeitos em equipamentos, recusa por parte de entrevistados em responder determinadas perguntas, entre outras.

Existem diversos métodos de tratamento para os dados faltantes, segundo ASSUNÇÃO (2012) o mais simples é utilizar medidas resumo como média, mediana e moda da coluna. Outro método, um pouco mais sofisticado consiste em realizar tais estimativas por uma regressão linear, algoritmos EM *(expectation-maximization)*, regressão multinomial, entre outras. Esse tratamento deve ser feito cuidadosamente para evitar distorções no resultado obtido.

2.1.2.2 Tratamento de outliers

De acordo com SILVA et al. (2004), os *outliers* são elementos que não seguem um padrão do conjunto de dados na qual pertencem, por exemplo uma idade mais avançada ao analisar expectativa de vida em uma região. A detecção dos *outliers*, pode revelar informações não esperadas e com certo grau de importância para algumas aplicações, como por exemplo: descoberta de fraudes. O autor assinala que os *outliers* ocorrem em pequenas proporções, pois se vários elementos fossem *outliers*, não seria possível caracterizar um padrão.

Existem alguns métodos para o tratamento de *outliers*. O método mais simples é a remoção desse valor, mas essa técnica deve ser usada somente quando o tamanho da base de dados é grande o suficiente. Há também a possibilidade de transformar os dados logaritmicamente, de modo que a variação causada pelos *outliers* seja diminuída.

2.1.2.3 Normalização de dados

Segundo PADILHA; CARVALHO (2017), há conjuntos de dados reais que apresentam atributos contínuos nas quais os valores de cada registro possuem várias faixas, devido às suas naturezas ou escalas que foram medidas, como, por exemplo, uma base de dados com atributos de pessoas, onde a coluna altura está em metros e a coluna peso em quilos. A variação da altura de uma pessoa com relação a outra é muito menor se comparada ao peso. Dessa forma, é necessário a normalização para que as medidas tenham a mesma ordem de grandeza.

Para normalizar os dados existem algumas técnicas. O método chamado de min-max consiste em subtrair o mínimo de cada valor e dividir esse resultado pela diferença entre o mínimo e máximo, o resultado será um valor entre 0 e 1. Já o método conhecido como *Standard* calcula a diferença entre os valores pela média. Em seguida, os valores são divididos pelo desvio padrão, de forma que a distribuição resultante tenha uma média 0 e desvio padrão igual a 1.

2.1.2.4 Seleção de atributos

A seleção de atributos busca identificar e remover colunas (ou atributos) irrelevantes e redundantes da base de dados. Atributos redundantes são aqueles fortemente correlacionados, que indicam um comportamento similar, por exemplo a distância percorrida e o consumo de combustível de um carro. Segundo SCHNEIDER (2018), o processo de seleção de atributos também é utilizado a fim de diminuir a dimensão dos dados analisados, de modo a auxiliar na velocidade e eficácia do aprendizado de máquina.

Para auxiliar nessa tomada de decisão, existem gráficos que exibem em pares a correlação entre os atributos, de forma a facilitar a percepção da correlação. Tendo encontrado a correlação, é preciso verificar seu impacto no resultado do modelo e analisar se é melhor mantê-la ou removê-la.

2.1.3 Modelos de aprendizagem de máquina

A aprendizagem de máquina em geral possui o objetivo de construir modelos computacionais que se adaptam e aprendem a partir da experiência.

Os algoritmos de aprendizagem de máquina buscam descobrir o relacionamento entre as variáveis de um sistema a partir de dados amostrados. Os algoritmos possuem diversas fontes de origem, como por exemplo, estatística, matemática, física, engenharia, entre outras.

Para este trabalho foi realizado testes prévios para visualização dos algoritmos com melhor desempenho, dessa forma usamos os algoritmos Regressão Logística, *Gradient Boosting* e Floresta Aleatória que iremos explicar a seguir.

2.1.3.1 Regressão linear

A regressão linear tem como objetivo traçar uma reta através dos dados em um diagrama de dispersão de modo a relacionar uma variável dependente a uma ou mais variáveis independentes. A regressão linear é dada simples quando possui uma variável independente e múltipla se possui mais de uma variável independente. A regressão linear pode ser utilizada em diversos âmbitos, como por exemplo nas relações simples, renda semanal e despesas, demanda dos produtos de uma empresa e a publicidade e na relação múltipla, por exemplo a qualidade de um produto químico que se relaciona com temperatura e pressão. Entretanto, é preciso estar ciente das limitações ao realizar a modelagem, pois por exemplo ao relacionar renda e idade, sabe-se que a renda tende a aumentar nas primeiras partes da idade adulta, achatar na vida adulta e por fim declinar ao aposentar. Portanto, a regressão por ter a forma linear não será capaz de descrever fielmente essa situação. A seguir será explicado a regressão linear simples e múltipla.

2.1.3.1.1 Regressão linear simples

A fim de entender o funcionamento da regressão linear simples, de acordo com RODRIGUES (2012), o modelo de regressão linear simples pode ser entendido como a relação linear entre a variável independente (X) e a variável dependente(Y).

A regressão apresentada na Equação (2.1) representa o modelo de regressão linear simples:

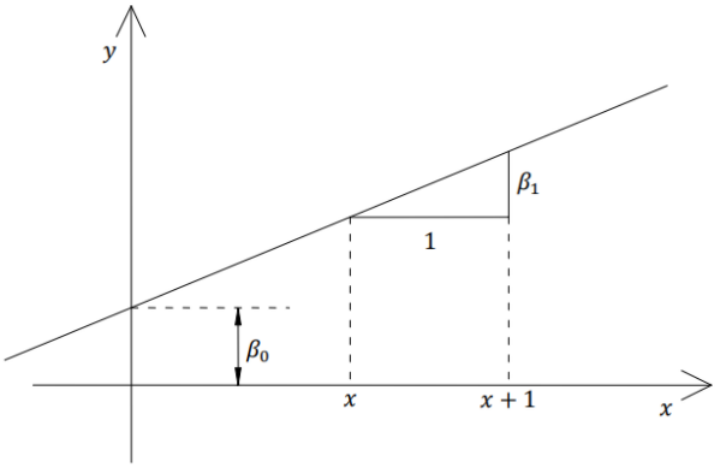
$$y_i = \beta_0 + \beta_1 x_1 + \varepsilon_i, \qquad i = 1, \cdots, n \tag{2.1}$$

onde:

- y_1 representa o valor da variável resposta ou dependente, Y, na observação i , $i = 1, \cdots, n$ (aleatória);
- x_1 representa o valor da variável independente, X, na observação i, $i = 1, \cdots, n$ (não aleatória);
- ε_i, $i = 1, \cdots, n$ são variáveis aleatórias que correspondem ao erro (variável que permite explicar a variabilidade existente em Y e que não é explicada por X);
- b_0 e b_1 correspondem aos parâmetros do modelo.

Na Figura 2.1, segundo RODRIGUES (2012) o coeficiente linear pode ser entendido como o ponto em que o parâmetro $\beta 0$ corta o eixo y quando x = 0. O parâmetro $\beta 1$ indica a inclinação da reta regressora, de modo que representa a mudança em Y, portanto indica que a cada unidade na variável X há a mudança na média de distribuição de probabilidade em Y.

Figura 2.1 – Interpretação geométrica dos parâmetros β_0 e β_1.



Fonte: RODRIGUES (2012)

2.1.3.1.2 Regressão linear múltipla

A fim de entender como a regressão logística funciona, é imprescindível entender os conceitos da regressão linear múltipla.

De acordo com RODRIGUES (2012), o modelo de regressão linear múltipla com p variáveis explicativas é definido da seguinte forma:

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \varepsilon_i, \qquad i = 1, \cdots, n \tag{2.2}$$

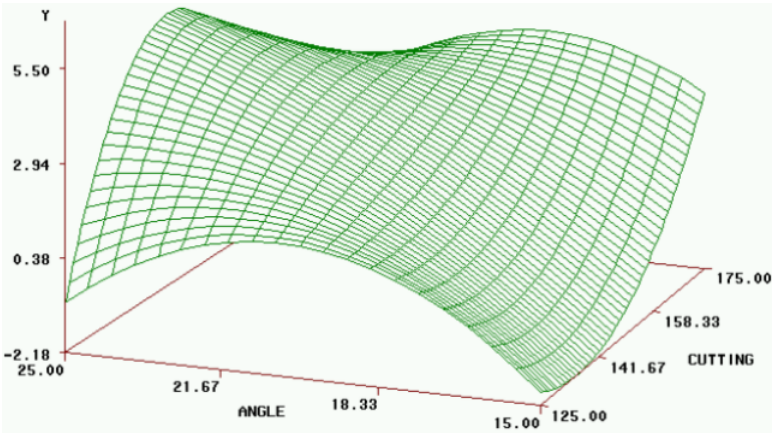
em que,

- y_1 representa o valor de variável resposta na observação i, $i = 1, \cdots, n$;
- $x_{i1}, x_{i2}, \cdots, x_{ip}$, $i = 1, \cdots, n$ são os valores da i-ésima observação das p variáveis explicativas (constantes conhecidas);
- $b_0, b_1, b_2, \cdots, b_p$ são os parâmetros ou coeficientes de regressão;

- ε_i, $i = 1, \cdots, n$, onde n corresponde aos erros aleatórios.

Este modelo devido a presença de mais de uma variável independente descreve um hiperplano que pode ser visto na Figura 2.2.

Figura 2.2 – Hiperplano p-dimensional referente às variáveis explicativas.



Fonte: RODRIGUES (2012)

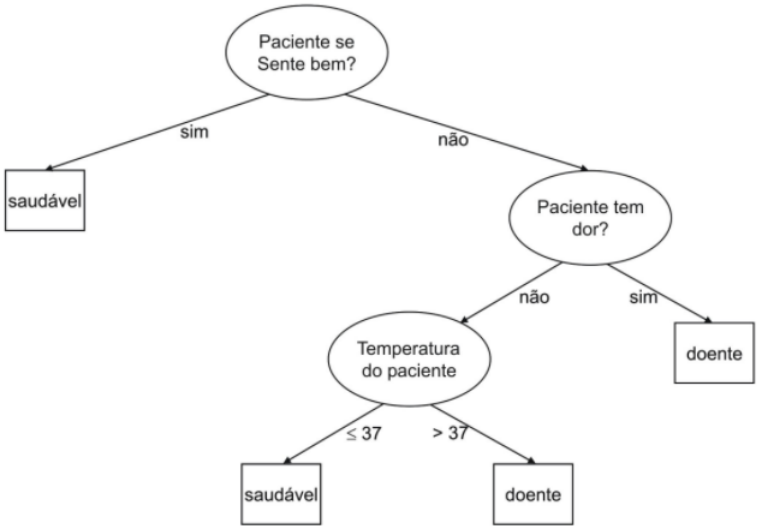
2.1.3.2 Árvore de decisão

A árvore de decisão é um algoritmo que a partir da formulação de regras de decisão simples é capaz de realizar uma previsão. O conjunto de regras de decisão pode ser facilmente visualizado no formato de árvore. Os atributos mais relevantes aparecem na parte superior da árvore, dessa forma possui uma fácil interpretação, entretanto pode se tornar complexa na determinação de uma condição de parada.

A Figura 2.3 demonstra um exemplo de árvore, em que a proposta é realizar o diagnóstico de um paciente. As elipses representam uma pergunta em um dos atributos e os retângulos apresentam pontos terminais, ou seja, a previsão.

O algoritmo da árvore de decisão propõe realizar diversas divisões dos dados em subconjuntos, de modo que os subconjuntos se tornem mais homogêneos. A homogeneidade é dada na medida em que os subconjuntos possuem menos classes ou apenas uma da variável alvo. A cada divisão calcula-se a entropia, que se-

Figura 2.3 – Árvore simples para diagnóstico de um paciente.



Fonte: MONARD; BARANAUSKAS (2003b)

gundo PRATES (2018) é responsável por determinar matematicamente o nível de "pureza" do subconjunto e o ganho de informação, a qual ocorre quando uma nova subdivisão dos dados provoca uma redução na entropia. Ao final do algoritmo é determinado os atributos mais relevantes na previsão.

2.1.3.3 Algoritmo Floresta Aleatória

O algoritmo Floresta Aleatória é do tipo *ensemble*, isso significa que é uma combinação de modelos de predição mais simples, em específico, árvores de decisão, de forma a criar um modelo mais robusto.

A estratégia denominada *Bagging* é utilizada nesse algoritmo, a qual, de acordo com Mayrink (2016), o algoritmo "sorteia aleatoriamente um subconjunto dos dados de treinamento e utiliza essa subamostra para treinar uma nova árvore de decisão. Em geral, o sorteio é feito com reposição , permitindo a ocorrência de observações replicadas nos subconjuntos de treinamento de cada modelo. (Mayrink, 2016)." Por exemplo, em uma base de dados para previsão de doença cardíaca,

há atributos como dor no peito, boa circulação sanguínea, artérias bloqueadas e peso. O primeiro passo será escolher aleatoriamente um subconjunto de registros dessa base. A partir desse conjunto será comparado dois atributos aleatoriamente e verificado qual possui menor entropia, de modo a escolher o mais relevante. A comparação entre atributos seguirá para a construção dos demais ramos. Dessa forma cria-se uma árvore aleatória, processo que será repetido várias vezes, construindo uma floresta.

Ao testar o algoritmo ou predizer uma nova entrada, cada árvore retorna sua resposta e por fim é feita a média ou um júri das respostas para previsão da resposta final.

O algoritmo Floresta Aleatória devido sua originalidade da árvore de decisão, também possui fácil interpretação, entretanto justamente por esse motivo, é preciso estar ciente da possibilidade de *overfitting* (modelagem se ajusta ao conjunto de dados) em alguns casos.

2.1.3.4 Algoritmo *Gradient Boosting*

O algoritmo *Gradient Boosting* possui algumas similaridades com o algoritmo Floresta Aleatória, pois ele também é do tipo ensemble e utiliza árvores de decisão como preditores mais simples. Entretanto, sua diferença em relação ao algoritmo Floresta Aleatória está na estratégia utilizada, a qual é denominada *Boosting*. Dessa forma, no algoritmo *Gradient Boosting*, MAYRINK (2015) pontua que segue-se o paradigma sequencial, a qual busca atuar sobre os erros obtidos na etapa anterior, com o objetivo que conforme novas árvores são criadas há a redução gradativa dos resíduos de previsão. O início do algoritmo é similar ao Floresta Aleatória, a qual se escolhe um subconjunto de registros e compara-se dois atributos várias vezes para a formação da árvore. A diferença entre os algoritmos se dá no prosseguimento, no algoritmo *Gradient Boosting* cada próxima árvore criada terá influência da anterior, pois será testado a cada árvore quais registros estão tendo

um maior erro e este terá um peso para aparecer na próxima árvore, de modo que a cada árvore criada o erro tende a diminuir. Por fim, cada árvore terá sua decisão e a previsão será calculada.

O algoritmo *Gradient Boosting* possui o benefício de diminuir o erro das previsões na construção das suas árvores, entretanto possui a fragilidade de seu modelo em alguns casos ser penalizado pelo *overfitting*, dessa forma é preciso estar atento a todas métricas de avaliação para determinar se é o algoritmo mais indicado em determinada situação.

2.1.3.5 Algoritmo regressão logística

O algoritmo de classificação conhecido como Regressão Logística é comumente utilizado para a classificação de problemas que possuem apenas duas classes para previsão, de forma que ela retorna a porcentagem de pertencimento de uma entrada àquela classe.

A regressão logística possui facilidades como a classificação de indivíduos em categorias e benefícios como o alto grau de confiabilidade, entretanto quando se trata de inteligibilidade, é mais difícil o compreender comparado a outros algoritmos. A seguir será explicado o funcionamento da regressão logística binária e multiclasse.

2.1.3.5.1 Algoritmo Regressão Logística Binária

Segundo MINUSSI; DAMACENA; JR (2002), na regressão logística, a chance de ocorrência de um evento pode ser estimada diretamente. A variável dependente Y assume apenas dois possíveis estados (0 ou 1) e há um conjunto de p variáveis independentes $X_1, X_2, \cdots, X_p$. o modelo de regressão logística pode ser escrito da seguinte forma:

$$P(Y = 1) = \frac{1}{1 + e^{-g(x)}} \tag{2.3}$$

onde,

$$g(x) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p \tag{2.4}$$

Os coeficientes $\beta_0, \beta_1, \cdots, \beta_p$ podem ser estimados a partir do conjunto de dados, por meio do método chamado máxima verossimilhança, a qual encontra uma combinação de coeficientes que maximiza a probabilidade de a amostra ter sido observada JR; LEMESHOW; STURDIVANT (2013). Dada combinação de coeficientes $\beta_0, \beta_1, \cdots, \beta_p$ e variando os valores de X, é possível notar que a curva logística tem comportamento probabilístico no formato da letra S, em que é característica da regressão logística. Esse formato agrega à regressão logística um alto grau de generalidade, aliada a aspectos muito desejáveis:

(a) Quando $g(x) \to +\infty$, então $P(Y = 1) \to 1$;

(b) Quando $g(x) \to -\infty$, então $P(Y = 1) \to 0$.

Assim como podemos estimar diretamente a probabilidade de ocorrência de um evento, podemos estimar a probabilidade de não ocorrência por diferença:

$$P(Y = 0) = 1 - P(Y = 1) \tag{2.5}$$

Ao utilizarmos a regressão logística, a principal suposição é a de que o logaritmo da razão entre as probabilidades de ocorrência e não ocorrência do evento é linear:

$$\frac{P(Y = 1)}{P(Y = 0)} = e^{\beta_0 + \beta_0 + \cdots + \beta_p} \tag{2.6}$$

e, por consequência,

$$\ln y \left[\frac{P(Y = 1)}{P(Y = 0)} \right] = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p \tag{2.7}$$

Por essa razão, ao interpretar os coeficientes da regressão logística, opta-se pela interpretação de e^B e não diretamente de B. Contudo, quando se utiliza o modelo logístico do ponto de vista de discriminação entre grupos, não há grande interesse na interpretação dos coeficientes GARSON (2000). Para utilizar o modelo de regressão logística para discriminação de dois grupos, a regra de classificação é a seguinte:

- se $P(Y = 1) > 0,5$ então classifica-se $Y = 1$;
- em caso contrário classifica-se $Y = 0$.

2.1.3.5.2 Algoritmo Regressão Logística Multiclasse

Para a regressão logística multiclasse, não é possível chegar ao resultado apenas aplicando o método conforme a regressão logística binária, pois ela retorna a possibilidade de um registro pertencer a uma classe. Dessa forma é preciso utilizar-se da abordagem "um contra todos", em que cada classe será por uma vez a de interesse, e as restantes a serem comparadas. Por exemplo, na base de dados de pétalas[1] , em que há três classes, Setosa, Virginica e Versicolor, cada classe será de interesse por um momento sendo 1 e as outras duas 0. Desse modo, será aplicado o modelo de regressão linear em cada análise que retorna os chamados logits, que servem de entrada para a função Softmax:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=i}^{k} e^{z_j}} \tag{2.8}$$

em que,

- e^{z_i} representa número de Euler elevado ao logit;
- $\sum_{j=i}^{k} e^{z_j}$ representa a somatória do número de Euler elevado aos logits.

[1] <www.kaggle.com/uciml/iris> acessado em 12/04/2021

2.1.4 Métricas de Avaliação de Modelos de Aprendizagem de Máquina

De acordo com SCHNEIDER (2018), existem diversas métricas de avaliação de modelos de aprendizagem de máquina, que são utilizadas a fim de estimar e comparar o resultado de diferentes classificadores, de forma a comparar a saída predita com a esperada.

Nesta seção, serão elucidadas as métricas utilizadas neste trabalho.

2.1.4.1 Acurácia

A acurácia apresentada na Equação (2.9) representa a proporção de registros classificados corretamente pela totalidade de registros previstos.

$$A = \frac{VP + VN}{VP + FP + VN + FN} \tag{2.9}$$

em que, A é a acurácia, VP os verdadeiros positivos (registros positivos que foram classificados como positivo), VN os verdadeiros negativos (registros negativos que foram classificados como negativo), FP os falsos positivos (registros negativos que foram classificados como positivos) e FN os falsos negativos (registros positivos que foram classificados como negativo).

A acurácia representa de forma simples o número de previsões corretas pelo número total de previsões, entretanto não é capaz de expor outras características das previsões, como a relação entre previsões positivas realizadas corretamente e todas as previsões que realmente são positivas. Dessa forma, é imprescindível a análise de outras métricas além da acurácia.

2.1.4.2 Acurácia pela validação cruzada

Segundo SCHNEIDER (2018), a validação cruzada é uma estimativa mais robusta do desempenho do classificador.

Para realizar o cálculo da acurácia pela validação cruzada, inicialmente o

> "conjunto de dados original é particionado aleatoriamente em k subconjuntos com dimensões idênticas ou semelhantes. Dentre os k subconjuntos, um é definido como o subconjunto de teste, enquanto que os demais são utilizados como os subconjuntos de treinamento. Este processo é repetido k vezes, sendo cada subconjunto utilizado apenas uma vez como dados de teste para a validação". SCHNEIDER (2018)

A média dos k resultados representa a acurácia pela validação cruzada.

2.1.4.3 Precisão

A precisão apresentada na Equação (2.10) representa a proporção de verdadeiros positivos pela soma dos verdadeiros positivos e falsos positivos.

$$P = \frac{VP}{VP + FP} \tag{2.10}$$

Para melhor entendimento, por exemplo na base de dados do Titanic [2], a precisão será calculada para cada classe, primeiro para a classe dos vivos e depois para os mortos. Desse modo, primeiro será calculada a proporção, dos passageiros classificados como vivos, quantos realmente ficaram vivos e por último a proporção, dos passageiros classificados como mortos, quantos realmente morreram. Nesse sentido, os falsos positivos possuem um peso importante no cálculo da precisão.

2.1.4.4 *Recall*

O *recall* apresentado na Equação (2.11) é a proporção dos verdadeiros positivos pela soma de verdadeiros positivos e falsos negativos.

$$R = \frac{VP}{VP + FN} \tag{2.11}$$

[2] <www.kaggle.com/c/titanic/overview> acessado em 12/04/2021

Para melhor entendimento, por exemplo na base de dados do Titanic, o *recall* será calculado para cada classe, primeiro para a classe dos vivos e depois para os mortos. Desse modo, primeiro será calculada a proporção, dos passageiros que ficaram vivos quantos foram classificados como vivos e por último a proporção, dos passageiros mortos, quantos foram classificados como mortos. Nesse sentido, os falsos negativos possuem um peso importante no cálculo do *recall*. Em geral, o entendimento de *recall* e precisão é muito similar, entretanto representam valores completamente diferentes.

2.1.4.5 F1-score

O F1-score apresentado na Equação (2.12) representa, de acordo com SAITO; REHMSMEIER (2015), é

> "uma média harmônica entre precisão e *recall*, refletindo em uma única medida o desempenho do modelo em predizer corretamente todas as instâncias verdadeiramente positivas e evitar a predição de falsos positivos". SAITO; REHMSMEIER (2015)

Portanto, o cálculo de f1-score informa em um único número a precisão e o *recall*.

$$F_1 = \frac{2PR}{P+R} \tag{2.12}$$

2.1.5 Trabalhos relacionados

Nesta seção serão apresentados trabalhos presentes na literatura que buscam prever resultados de jogos de futebol e que se relacionam com os objetivos deste trabalho. SCHNEIDER (2018) observou que uma tarefa difícil no âmbito das apostas esportivas é predizer o resultado final de partidas de futebol. Desse modo, ele buscou discutir alternativas nesse sentido, de modo a comparar o desempenho de dez classificadores, variando os atributos, hiperparâmetros e a quantidade de

partidas utilizadas com o objetivo de encontrar o melhor classificador para esse tipo de trabalho.

Os atributos foram escolhidos conforme o uso na literatura, e baseou-se em características pré-jogo dos times envolvidos, tais como o *ELO rating*(ranking de um time), formação inicial dos jogadores, probabilidades de Poisson, forças ofensiva e defensiva (calculado baseado nos gols feitos e sofridos dentro e fora de casa de um time), entre outras.

Para a escolha dos hiperparâmetros foram analisadas as propriedades acurácia e F1-Score conforme a variação dos múltiplos classificadores. Por fim, a quantidade de partidas utilizadas foi variada, ignorando-se 3, 10 ou 19 partidas de um início de cada temporada. Essa estratégia foi utilizada, pois atributos como a distribuição de Poisson para ser calculado com uma maior precisão, precisam de um maior número de partidas para contabilização.

Dessa forma, analisando-se os vários classificadores foi possível verificar que aqueles que obtiveram maior acurácia foram a Regressão Logística (54.73%), a Análise Discriminante Linear (54.82%), SVM (54.35%) e o K-vizinhos mais próximos (54.34%). Além de classificadores individuais, foi utilizado um classificador ensemble composto de seis classificadores que apresentou acurácia de 57%.

SCHMIDT (2017) realizou o desenvolvimento de um sistema para auxiliar na previsão do resultado de uma partida de futebol, de forma que a saída seja o resultado com maior chance de ocorrência. O autor utilizou dados estatísticos de partidas do campeonato Inglês e dados de habilidade dos jogadores do jogo virtual Fifa.

As técnicas de classificação Floresta Aleatória, Support Vector Machine e Redes Neurais foram testadas para determinar o resultado final da partida, e o melhor aproveitamento obtido foi de 58,77% com a técnica Redes Neurais.

NABINGER (2018) utilizou algoritmos de aprendizagem supervisionada, e detectou quais variáveis são as mais importantes para determinar a vitória em uma partida a partir de 64 jogos da Copa do Mundo de 2018.

Foram analisados 40 atributos para esse estudo, divididos em 4 grupos: Ataque, Performance, Defesa e Disciplina. Um dos atributos mais comentados, a posse de bola, não possui forte relação com a vitória de um jogo. Um exemplo foi o jogo entre Rússia e Espanha, na qual a Espanha teve 75

A partir de técnicas do aprendizado supervisionado foi possível identificar que a variável de Ataque **chutes no gol** e as variáveis defensivas **desarmes** e **chutes bloqueados** foram as que mais contribuíram para a vitória de um time com um acerto de 57,81%.

Em nosso trabalho aplicamos os algoritmos *Gradient Boosting*, Regressão Logística e Floresta Aleatória para prever partidas de futebol de uma base de dados contendo partidas dos campeonatos, Brasileiro Séries A e B, Libertadores, Bundesliga, La Liga, Premier League, Serie A Tim, Liga dos Campeões e Liga Europa. A forma de coleta e análise dos dados, bem como de criação do modelo será apresentado a seguir, no próximo capítulo.

3 METODOLOGIA

Este capítulo irá apresentar ao leitor as ferramentas e técnicas utilizadas para o estudo proposto. O trabalho aqui apresentado visa predizer o resultado final de uma partida de futebol a partir das informações do primeiro tempo da partida, ou seja, no intervalo.

3.1 Web scraping e características dos dados

Os dados foram coletados do site Flash Score[1]. O site Flash Score disponibiliza as principais estatísticas de diversos campeonatos que acontecem no mundo. Por meio dele é possível acompanhar estatísticas ao vivo e também de jogos que já ocorreram. Para os jogos já realizados é possível visualizar as estatísticas finais do jogo e também apenas do primeiro tempo, na qual possibilitou a coleta de um maior volume de dados em um tempo curto. Para elaboração da parte prática desse trabalho foi seguido os passos abaixo:

- Coleta dos dados.
- Tratamento dos dados.
- Treinamento, teste e avaliação dos resultados.
- Criação do sistema para auxiliar os usuários.

A Figura 3.1 mostra a página principal do site.

Para coletar os dados foi construído um script em Python[2] usando as bibliotecas Requests[3], BeautifulSoup[4] e Selenium[5] do Python.

[1] <www.flashscore.com.br> acessado em 12/04/2021
[2] <https://docs.python.org/3/>, versão 3.8.5, acessado em 12/04/2021
[3] <https://pypi.org/project/requests/>, versão 2.22.0, acessado em 12/04/2021
[4] <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>, versão 4.9.1 ,acessado em 12/04/2021
[5] <https://selenium-python.readthedocs.io/>, versão 3.141.0, acessado em 12/04/2021

Figura 3.1 – Site “Flashscore”.

O algoritmo de funcionamento do web scraper é demonstrado no fluxo abaixo.

1. Abrir página com os jogos do campeonato
2. Identificar link de cada jogo que contém as estatísticas
3. Coletar as estatísticas de cada partida
4. Escrever no arquivo cada jogo

A coleta de dados segue basicamente esses passos, de modo que o processo é reiniciado manualmente após o término da coleta de uma temporada de um campeonato. As Figuras 3.2 e 3.3 demonstram as páginas abertas durante o processo e as respectivas informações encontradas.

A Tabela 3.1 mostra os atributos coletados do site diretamente, bem como o tipo e descrição de cada campo.

Atributo	**Tipo**	**Descrição**
pais	String	Informa o país ou região a qual os times pertencem.
campeonato	String	Informa o nome do campeonato disputado.

Atributo	**Tipo**	**Descrição**
timeMandante	String	Informa o nome do time com mando de campo.
timeVisitante	String	Informa o nome do time visitante.
posseBolaM	Int	Informa o valor da posse de bola do time mandante.
posseBolaV	Int	Informa o valor da posse de bola do time visitante.
tentativasGolM	Int	Informa a quantidade de vezes que o time mandante tentou finalizar ao gol do time adversário (soma das finalizações, chutes fora e chutes bloqueados).
tentativasGolV	Int	Informa a quantidade de vezes que o time visitante tentou finalizar ao gol do time adversário.
finalizacoesM	Int	Informa a quantidade de finalizações (chutes a gol) do time mandante.
finalizacoesV	Int	Informa a quantidade de finalizações do time visitante.
chutesForaM	Int	Informa a quantidade de chutes realizados que não foram em direção ao gol pelo time mandante.
chutesForaV	Int	Informa a quantidade de chutes realizados que não foram em direção ao gol pelo time visitante.
chutesBloqueadosM	Int	Informa a quantidade de tentativas de finalizações que foram bloqueadas pelo time mandante (defensores).
chutesBloqueadosV	Int	Informa a quantidade de tentativas de finalizações que foram bloqueadas pelo time visitante.
faltasCobradasM	Int	Informa a quantidade de faltas sem contato (Exemplo: bola na mão) sofridas pelo time mandante.
faltasCobradasV	Int	Informa a quantidade de faltas sem contato sofridas pelo time visitante.
escanteiosM	Int	Informa a quantidade de escanteios cedidos ao time mandante.

Atributo	Tipo	Descrição
escanteiosV	Int	Informa a quantidade de escanteios cedidos ao time visitante.
impedimentosM	Int	Informa a quantidade de impedimentos feitos pelo time mandante.
impedimentosV	Int	Informa a quantidade de impedimentos feitos pelo time visitante.
defesasGoleiroM	Int	Informa a quantidade de intervenções realizadas por bolas que foram em direção ao gol pelo goleiro do time mandante.
defesasGoleiroV	Int	Informa a quantidade de intervenções realizadas por bolas que foram em direção ao gol pelo goleiro do time visitante.
faltasM	Int	Informa a quantidade de faltas com contato sofridas pelo time mandante.
faltasV	Int	Informa a quantidade de faltas com contato sofridas pelo time visitante.
cartoesAmarelosM	Int	Informa a quantidade de cartões amarelos dados ao time mandante.
cartoesAmarelosV	Int	Informa a quantidade de cartões amarelos dados ao time mandante.
cartoesVermelhosM	Int	Informa a quantidade de cartões vermelhos dados ao time mandante.
cartoesVermelhosV	Int	Informa a quantidade de cartões vermelhos dados ao time mandante.
totalPassesM	Int	Informa a quantidade de passes realizados pelo time mandante.
totalPassesV	Int	Informa a quantidade de passes realizados pelo time visitante.

Atributo	Tipo	Descrição
desarmesM	Int	Informa a quantidade de desarmes realizados pelo time mandante.
desarmesV	Int	Informa a quantidade de desarmes realizados pelo time visitante.
ataquesM	Int	Informa a quantidade de jogadas iniciadas pelo time mandante.
ataquesV	Int	Informa a quantidade de jogadas iniciadas pelo time visitante.
ataquesPerigososM	Int	Informa a quantidade de vezes que o time mandante teve a posse de bola no campo de ataque.
ataquesPerigososV	Int	Informa a quantidade de vezes que o time visitante teve a posse de bola no campo de ataque.
diferencaGols	Int	Informa o placar do jogo ao final do primeiro tempo (Exemplo: Jogo a qual está 1x4 para o time visitante será denotado por -3).
placarPrimeiroTempo	Char	Informa o placar do jogo ao final do primeiro tempo, em que possui 3 valores possíveis, ‘V’ para vitória do mandante, ‘E’ para empate e ‘D’ para vitória do visitante.
placarFinal	Char	Informa o placar do jogo ao final, em que possui 3 valores possíveis, ‘V’ para vitória do mandante, ‘E’ para empate e ‘D’ para vitória do visitante.

Tabela 3.1 – Atributos coletados pelo web scraper

A Tabela 3.2 mostra os valores dos atributos, tais como a quantidade de entidades coletadas, e a média e desvio padrão para variáveis quantitativas.

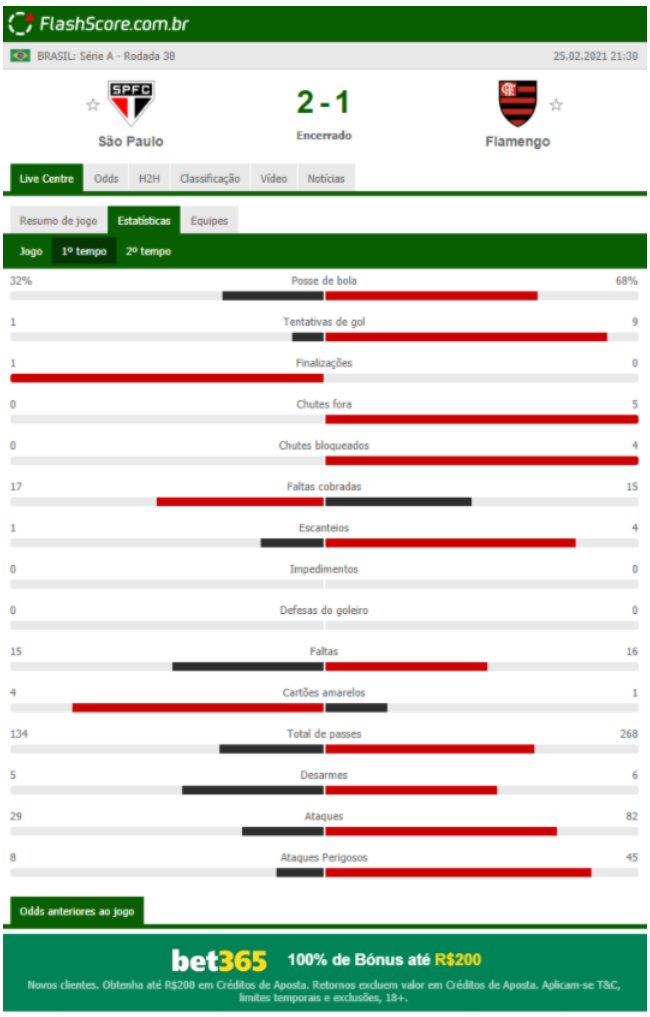
Figura 3.2 – Jogos da temporada 2020 do campeonato Brasileiro Série A.

Atributo	**Quantidade de entidades coletadas**	**Média**	**Desvio padrão**
pais	18994	-	-
campeonato	18994	-	-
timeMandante	18994	-	-
timeVisitante	18994	-	-
posseBolaM	16703	51,75	10,65
posseBolaV	16703	48,21	10,64
tentativasGolM	17219	6,28	3,1
tentativasGolV	17219	4,82	2,71
finalizacoesM	17220	2,33	1,71
finalizacoesV	17220	1,78	1,48
chutesForaM	17134	2,75	1,81
chutesForaV	17134	2,11	1,57
chutesBloqueadosM	13565	1,53	1,41
chutesBloqueadosV	13565	1,18	1,22
faltasCobradasM	14238	7,43	2,92
faltasCobradasV	14238	7,38	2,89

Atributo	**Quantidade de entidades coletadas**	**Média**	**Desvio padrão**
escanteiosM	17138	2,73	1,9
escanteiosV	17138	2,06	1,63
impedimentosM	17069	1,1	1,16
impedimentosV	17069	0,98	1,1
defesasGoleiroM	17215	1,28	1,26
defesasGoleiroV	17215	1,65	1,46
faltasM	17111	6,43	2,71
faltasV	17111	6,63	2,77
cartoesAmarelosM	19013	0,73	0,83
cartoesAmarelosV	19013	0,89	0,89
cartoesVermelhosM	19013	0,01	0,13
cartoesVermelhosV	19013	0,02	0,16
totalPassesM	4961	237,76	72,96
totalPassesV	4961	220,14	70,76
desarmesM	4110	7,57	3,29
desarmesV	4110	7,76	3,36
ataquesM	6630	53,46	17,35
ataquesV	6630	48,94	16,25
ataquesPerigososM	6631	27,51	12,55
ataquesPerigososV	6631	22,13	10,72
diferencaGols	19013	0,17	1,09
placarPrimeiroTempo	19013	-	-
placarFinal	19013	-	-

Tabela 3.2 – Características dos dados coletados.

Figura 3.3 – Estatísticas do primeiro tempo no jogo São Paulo e Flamengo pela rodada 38



3.2 Tratamento dos dados

Nesta etapa foi utilizada a linguagem Python para manipulação e foi necessária a biblioteca Pandas[6] para o tratamento dos dados da Tabela 3.1. Primeiramente, foi realizada a remoção de colunas que a priori por uma percepção inicial, não possuem influência significativo no resultado ou trariam uma grande complexidade na criação do modelo, por exemplo, ‘pais’, ‘campeonato’, ‘timeMandante’,

[6] <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html>, versão 1.1.0, acessado em 12/04/2021

‘timeVisitante’. Além dessas colunas, também foram removidas as colunas ‘chutesBloqueadosM’, ‘chutesBloqueadosF’, ’Chutes foraM’ e ’Chutes foraV’, pois os atributos ‘finalizacoesM’, ‘finalizacoesV’, ‘tentativasGolM’ e ‘tentativasGolV’ já representam a estatística de quantas vezes o time chegou perto do gol adversário.

Após a remoção de algumas colunas, foi necessário tratar os valores nulos presentes na base de dados. Para isso, os valores faltantes foram preenchidos usando a média da coluna correspondente.

O próximo passo foi identificar possíveis *outliers* e removê-los. Dessa forma, para cada coluna foram verificados os valores únicos presentes naquela coluna, por exemplo, na posse de bola de mandante foram encontrados valores como, 46, 48, 50. Dessa forma, foi possível visualizar quais dados estavam se destoando dos demais. Valores anômalos foram detectados e suas entidades removidas completamente. Isso ocorreu nas seguintes situações:

- O valor -1 foi encontrado nas colunas ‘defesasGoleiroM’ e ‘defesasGoleiroV’.
- O valor 0 encontrado nas colunas ‘totalPassesM’ e ‘totalPassesV”.

Nessa etapa a fim de diminuir a quantidade de atributos de entrada e padronizá-los, foi realizada a diferença entre os valores que possuem a característica mandante e visitante, por exemplo, o atributo de finalizações foi calculado a partir da diferença entre ‘finalizacoesM’ e ‘finalizacoesV’. Dessa forma um time mandante que finalizou 10 vezes contra 6 do time visitante terá na coluna ‘finalizacoes’ o valor de 4. Após o cálculo da nova coluna, foram removidas as colunas participantes. Além disso, o atributo ‘faltas’ foi calculado como sendo a soma de ‘faltasM’ e ‘faltasCobradasM’ menos ‘faltasV’ e ‘faltasCobradasV’.

Foi preciso tratar a coluna ‘placarPrimeiroTempo’ separadamente, pois apresenta valores categóricos como ‘V’, ‘D’ e ‘E’. Dessa forma, foi usada a estratégia de variáveis ‘dummies’. Nesta coluna há três valores possíveis, dessa

maneira foram criadas duas variáveis dummy responsáveis por representar esses 3 valores. Portanto, se o valor for ‘D’, será representado por 0 e 0, se for ‘E’ será representado por 0 e 1 e por fim ‘V’ representado por 1 e 1. Tendo realizado esse processo, a coluna inicial foi removida.

A fim de normalizar os dados, foi utilizada a estratégia denominada ‘MinMaxScaler’, em que a re-escala dos dados é feita de forma independente entre cada coluna. Desse modo, as colunas numéricas passaram a ter um valor entre -1 e 1. Foi testado o resultado com normalização e sem, e com a normalização apresentou um resultado melhor.

As classes estavam desbalanceadas na base de dados original, tendo 8899 registros de vitórias, 5418 registros de derrotas e 4684 registros de empates. Sabe-se que o mandante possui certa vantagem por conhecer melhor as condições do campo de jogo e pela presença da torcida. Esta vantagem pôde ser observada na distribuição das classes, em que as vitórias do mandante representam 46,8%, empates 24,7% e derrotas do mandante 28,5%. Entretanto, para manter um resultado mais fiel às estatísticas, foi balanceado as classes de forma que as classes vitória e derrota se igualaram a quantidade de registros de empate, que são 4684 cada. Para esse balanceamento foi utilizada a instância NearMiss da biblioteca imblearn[7]. Por fim, foi realizada a remoção de colunas que apresentam correlação direta e foram confirmadas pelos dados, por exemplo, a posse de bola possui forte correlação com o número de passes, dessa forma foi removida a coluna ‘totalPasses’. Isso também ocorreu para a correlação entre finalizações e defesas do goleiro, na qual foi removida a coluna ‘defesasGoleiro’.

3.3 Aprendizagem de máquina

Para a separação dos dados em treino e teste foi utilizado 75% e 25% respectivamente.

[7] <https://pypi.org/project/imblearn/>,versão 0.7.0, acessado em 12/04/2021

A instanciação dos modelos e a avaliação posterior foi realizada a partir da biblioteca sklearn. As bibliotecas matplotlib[8] e seaborn[9] foram usadas para a visualização da matriz de confusão e importância dos atributos.

Após o aprendizado foi utilizada a técnica K-fold para validação cruzada, que realiza diversas divisões nos dados para treino e teste a fim de evitar problemas de aleatoriedade e assim chegar a uma acurácia mais próxima da realidade. Ao realizar esse procedimento, foi utilizado o valor padrão para as divisões a serem feitas (cv), portanto a divisão entre treino e teste foi feita cinco vezes de forma a utilizar uma porção para teste e quatro para treino. O resultado, dessa forma, é dado pela média das acurácias.

As métricas empregadas para medir os resultados foram acurácia, a média das acurácias dada pela validação cruzada, f1-score, precisão e *recall*.

[8] <https://matplotlib.org/>, versão 3.3.0, acessado em 12/04/2021
[9] <https://seaborn.pydata.org/>, versão 0.10.1, acessado em 12/04/2021

4 RESULTADOS

Este capítulo irá apresentar os resultados obtidos após o treinamento e teste dos algoritmos apresentados anteriormente.

4.1 Classificação em três classes: vitória, empate e derrota

A Tabela 4.1 mostra a acurácia, acurácia pela validação cruzada, f1-score, precisão e *recall* referente aos algoritmos *Gradient Boosting*, Regressão Logística e Floresta Aleatória, usados neste trabalho.

Tabela 4.1 – Métricas dos algoritmos testados.

Algoritmo	**Acurácia %**	**Acurácia (validação cruzada %)**	**F1-score (D,E,V) %**	**Precisão (D,E,V) %**	**Recall (D,E,V) %**
Gradient Boosting	60,34	55,18	[64,16; 50,9;66,16]	[67,99;50,2, ;63,7]	[60,74 ;51,61; 68,83]
Regressão Logística	57,42	57,25	[63,79; 48,28; 61,69]	[68,00; 44,23; 64,58]	[60,06; 53,14; 59,04]
Floresta Aleatória	55,62	50,27	[60,36; 43,98 ;62,6]	[60,23; 47,8; ,58,61]	[60,49; 40,73; ,67,18]

A partir da Tabela 4.1, é possível verificar que os algoritmos apresentaram uma acurácia similar, sendo que o algoritmo *Gradient Boosting* mostrou-se o mais acurado. Entretanto, após realizar a validação cruzada, o algoritmo Regressão Logística conseguiu manter sua acurácia próxima à calculada inicialmente e teve um desempenho melhor que os outros dois algoritmos, com uma acurácia de 57,25%.

A Figura 4.1 apresenta a matriz de confusão do algoritmo *Gradient Boosting*.

A partir da matriz de confusão a qual foi usado o algoritmo *Gradient Boosting*, mostrado na Figura 4.1, é possível extrair as métricas, precisão, *recall* e f1-score. Nota-se, que jogos envolvendo empate (colunas 01, 10, 12, 21) possuem um certo grau de erro, de forma a atrapalhar a acurácia do modelo.

A Figura 4.2 mostra a importância dos atributos encontrados no algoritmo *Gradient Boosting*.

Em relação ao modelo usando o *Gradient Boosting*, a principal característica para determinar a classe pertencente a um novo registro foi a diferença de gols.

Figura 4.1 – Matriz de confusão do algoritmo Gradient Boosting. 0 (D) 1 (E) 2(V).

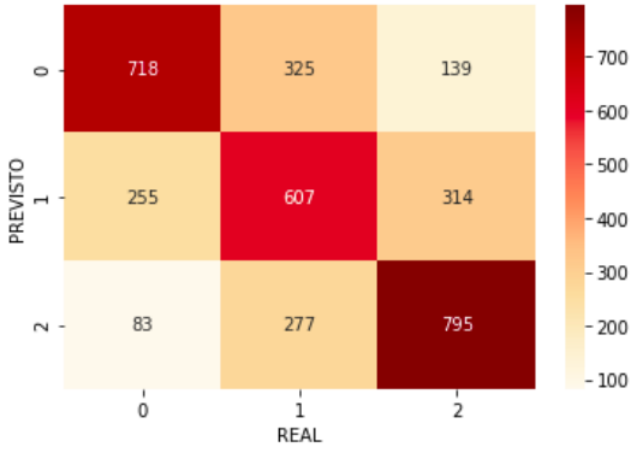
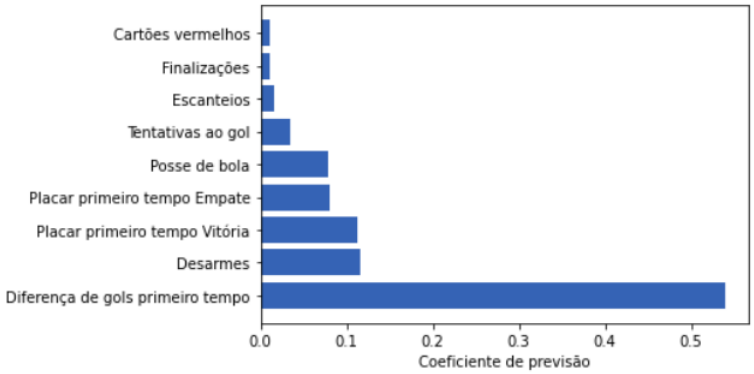


Figura 4.2 – Importância dos atributos do algoritmo Gradient Boosting



Portanto, se um dos times estiver vencendo a partida, dificilmente o algoritmo irá prever uma possível virada. Em segundo lugar, aparece o número de desarmes. Outras características apresentam um coeficiente de previsão menor, como posse de bola, finalizações, etc.

A Figura 4.3 mostra que assim como o algoritmo *Gradient Boosting* visto anteriormente, a matriz de confusão do algoritmo Regressão Logística possui as células que representam acertos na previsão como maioria, entretanto os jogos envolvendo empate dificultaram o modelo a identificar a classe correta pertencente.

Figura 4.3 – Matriz de confusão do algoritmo Regressão Logística. 0 (D) 1 (E) 2(V).

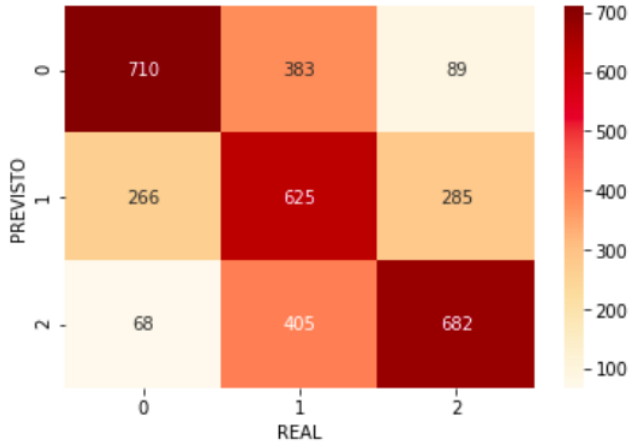
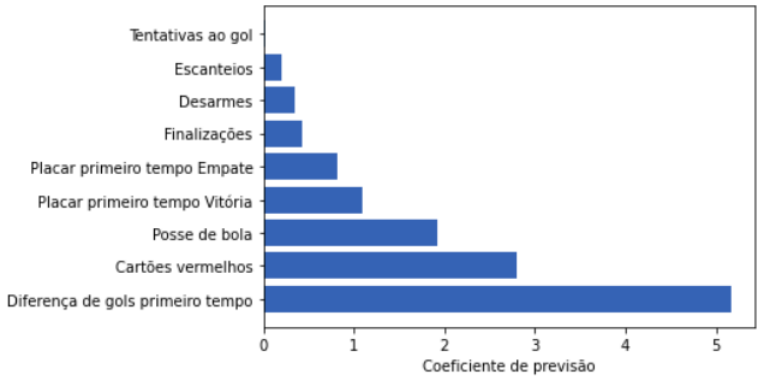
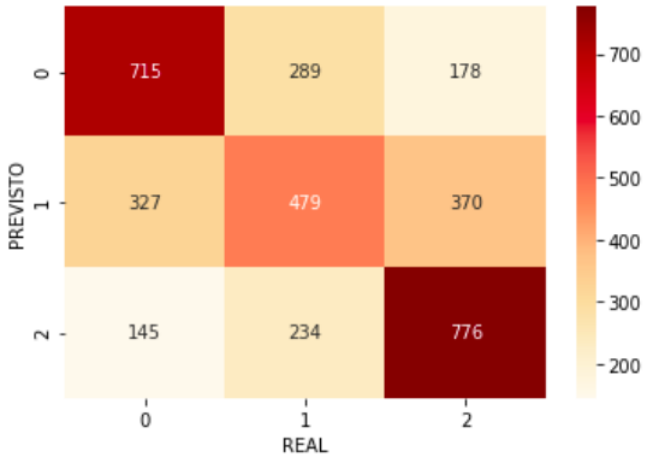


Figura 4.4 – Importância dos atributos do algoritmo Regressão Logística



A Figura 4.4 mostra que a importância dos atributos calculada pelo algoritmo Regressão Logística é similar ao algoritmo *Gradient Boosting* referente ao atributo que possui maior relevância no momento da predição. Ambos os algoritmos apontam a diferença de gols como determinante para o resultado final da partida. O cartão vermelho diagnosticou-se um importante atributo a se considerar também. A posse de bola, apresentada em trabalhos anteriores como não determinante para o resultado final, mostrou-se um fator importante para este algoritmo, mas que não aponta com certeza o resultado final.

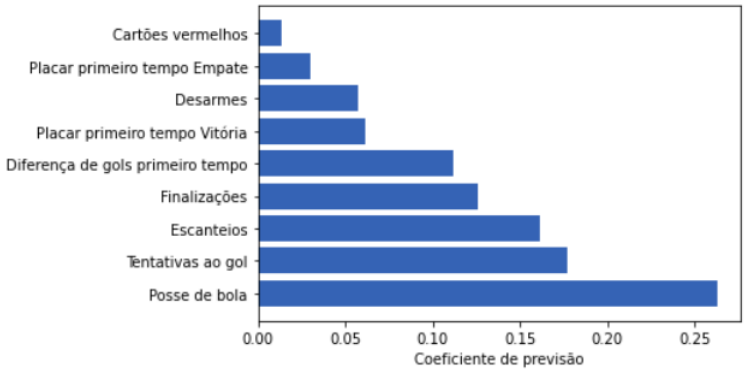
Figura 4.5 – Matriz de confusão do algoritmo Floresta Aleatória



. 0 (D) 1 (E) 2(V).

A Figura 4.5 mostra a matriz de confusão do algoritmo Floresta Aleatória. Nota-se um resultado similar aos algoritmos apresentados anteriormente, pois a predição envolvendo empate há uma maior taxa de erros, reduzindo assim a acurácia.

Figura 4.6 – Importância dos atributos do algoritmo Floresta Aleatória



Ao contrário dos algoritmos apresentados anteriormente e do trabalho relacionado (Nabinger, 2018), o algoritmo Floresta Aleatória determinou que a posse de bola é um atributo com extrema importância. Provavelmente por isso o algoritmo Floresta Aleatória apresentou uma acurácia menor pela validação cruzada

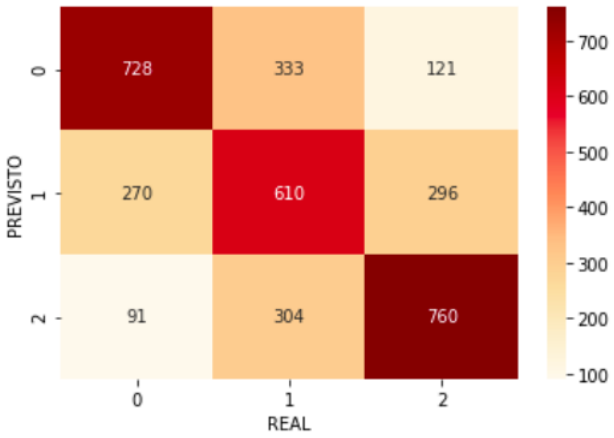
de cerca de 7% de diferença com relação ao algoritmo Regressão Logística. O júri abaixo foi criado baseado nos algoritmos *Gradient Boosting*, Regressão Logística e Floresta Aleatória. Os resultados estão na Tabela 4.2 e Figura 4.7.

Tabela 4.2 – Métricas do júri.

Algoritmo	Júri
Acurácia %	59,72
Acurácia (validação cruzada %)	55,26
F1-score (D,E,V) %	[64,11; ,50,35; 65,18]
Precisão (D,E,V) %	[66,85; 48,91; 64,57]
Recall (D,E,V) %	[0.61;,0.51; ,0.65;]

O júri apresentou um resultado insatisfatório, pois obteve uma acurácia pela validação cruzada de 55,26%, próxima ao de um modelo individualmente.

Figura 4.7 – Matriz de confusão do Júri 0 (D) 1 (E) 2(V)



Comparado aos modelos demonstrados anteriormente, o algoritmo representado pelo júri também apresentou um resultado similar pela dificuldade em lidar com os empates.

Dos algoritmos analisados, o algoritmo Regressão Logística apresentou melhor desempenho, com a acurácia pela validação cruzada sendo superior aos demais. Apesar de o algoritmo Gradient Boosting apresentar métricas como f1-score superior, acredita-se que ocorreu overfitting por conta que as árvores de decisão

trazem consigo esse tipo de problema, a qual é multiplicado no algoritmo Gradient Boosting. Como explicado anteriormente na seção de modelos de algoritmos de máquina, é um problema comum neste tipo de modelo, que foi visto ao utilizar a validação cruzada, de modo que sua acurácia foi diminuída substancialmente.

Portanto, nesse momento chega-se a resposta dos objetivos iniciais que eram conseguir as melhores métricas na predição do resultado final e identificar as variáveis com maior importância nessa previsão. Desse modo as melhores métricas foi obtida pelo algoritmo Regressão Logística e as variáveis com maior peso ao intervalo de um jogo no resultado final de uma partida são a diferença de gols no placar, cartão vermelho, posse de bola, situação do placar do jogo, finalizações, desarmes e escanteios.

4.2 Classificação usando duas classes: vitória e derrota

Considerando os piores resultados obtidos em jogos envolvendo empates, buscamos reduzir o número de classes de classificação para duas, a fim de compreender o que acontece quando o empate é eliminado do modelo. Estes resultados são úteis para compreender como se comportaria o processo de predição neste novo cenário, e são úteis para fins didáticos de análise e investigação. Na prática, remover o empate das classes certamente não seria aceitável em um sistema de predição de resultados de jogos de futebol.

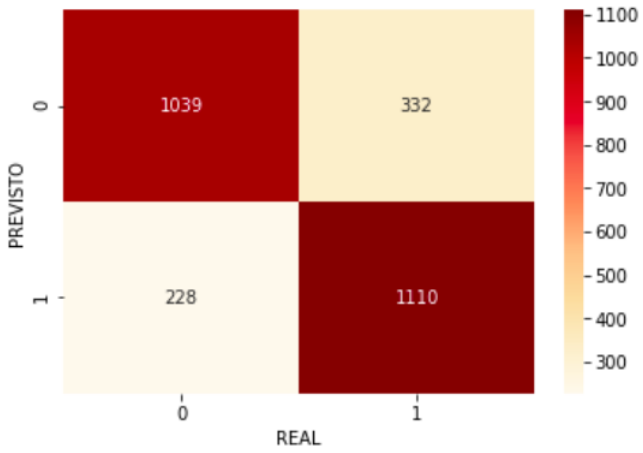
A Tabela 4.3 apresenta os resultados obtidos pelo algoritmo Regressão Logística ao utilizar apenas duas classes.

Tabela 4.3 – Métricas do algoritmo Regressão Logística em duas classes.

Algoritmo	Regressão Logística
Acurácia %	79,33
Acurácia (validação cruzada) %	77,14
F1-score (D,V) %	[78,77; 79,85]
Precisão (D,V) %	[82,00;76,97]
Recall (D,V) %	[75,78;82,95]

É notória a melhora na acurácia em comparação aos resultados apresentados na seção anterior, com três classes. Neste cenário, o algoritmo recebeu como entrada partidas que estavam em resultado parcial vitória do mandante, empate e derrota do mandante e retornou apenas duas classes, sendo vitória ou derrota do mandante. A Figura 4.8 mostra a matriz de confusão retornada com o uso do algoritmo Regressão Logística com duas classes.

Figura 4.8 – Matriz de confusão do algoritmo Regressão Logística para duas classes 0 (D) 1(V)

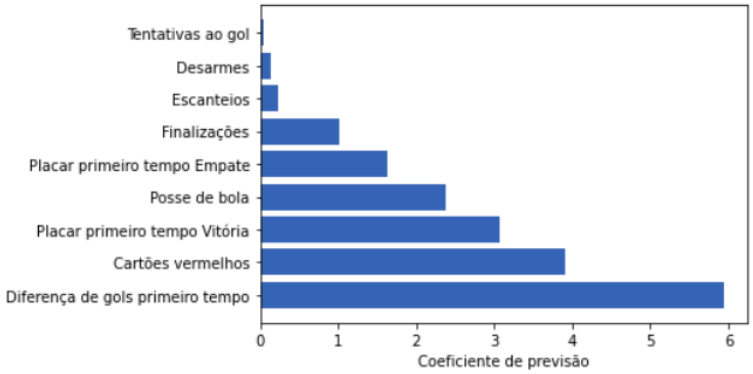


Nota-se que a matriz de confusão apresenta um ótimo resultado quando utilizado somente duas classes para previsão. Ao excluir os empates, a dificuldade em decidir a qual classe pertence o registro diminuiu de modo significativo.

A Figura 4.9 mostra a importância dos atributos obtidos pelo algoritmo Regressão Logística com duas classes.

A importância dos atributos ficou semelhante à vista com a utilização de três classes, ou seja, esses atributos realmente apresentam um coeficiente de previsão favorável para a previsão do resultado final.

Figura 4.9 – Importância dos atributos do algoritmo Regressão Logística para duas classes



4.3 Sistema

Foi criada uma interface gráfica web para que usuários possam utilizar esse serviço de forma simples. Nesta interface, os jogos ao vivo em andamento são exibidos, e o usuário pode selecionar um jogo que tenha interesse em saber o resultado final. O jogo necessita estar no intervalo e ter as estatísticas disponíveis para que seja previsto.A Figura 4.10 apresenta os jogos ao vivo em um determinado momento.

A Figura 4.11 apresenta a previsão realizada pelo algoritmo baseado nas estatísticas do primeiro tempo. A probabilidade de cada classe é obtida por meio do método *predict_proba*.

Figura 4.10 – Jogos ao vivo

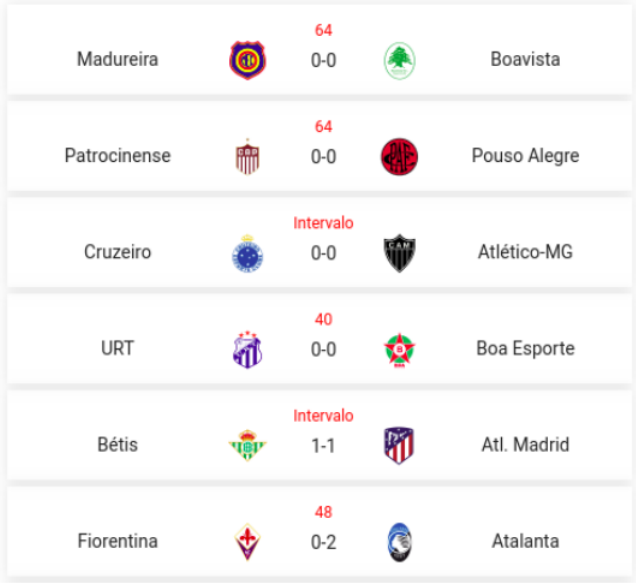


Figura 4.11 – Previsão do jogo entre Bétis e Atlético de Madrid

5 DISCUSSÃO E CONCLUSÕES

Este trabalho teve como objetivo analisar partidas de futebol a partir de estatísticas do primeiro tempo e criar um modelo de forma a predizer o resultado final das partidas. Os resultados mostraram que a melhor estratégia obtida para a predição foi a utilização do algoritmo Regressão Logística para três classes de classificação: vitória, empate e derrota.

Em relação aos trabalhos já feitos na área, o nosso trabalho não considera os dados pré-jogo que podem favorecer um dos times durante o jogo, o que pode ter afetado a acurácia obtida. Em contrapartida, nosso trabalho faz as análises em tempo real durante a partida, o que se torna um produto interessante.

Um dos motivos pelos quais acreditamos que nosso trabalho obteve os resultados que observamos foi pela quantidade de entidades coletadas, em boa quantidade. O modelo, porém, poderia ser aperfeiçoado com a adição de mais entidades na base. Além disso, foi detectado que atributos já mencionados em trabalhos anteriores, como o número de desarmes no algoritmo *Gradient Boosting* permaneceram com seu coeficiente de previsão favorável.

O tratamento dos dados[1], bem como o código-fonte da coleta[2], API[3] e o sistema web[4] estão disponíveis.

Este trabalho apresenta algumas fragilidades, como a dificuldade em trabalhar com 3 classes, na qual o empate impacta de forma abrangente nos resultados do modelo. Além disso, com a não existência de dados pré-jogo, e de dados ao vivo como lesão de jogador, espera-se que previsões de viradas de jogo não sejam realizadas.

[1] <https://github.com/heullvers/tccjupy> acessado em 12/04/2021
[2] <https://github.com/heullvers/firstimeColeta> acessado em 12/04/2021
[3] <https://github.com/heullvers/firstimeapi> acessado em 12/04/2021
[4] <https://github.com/heullvers/pagevuehalftime> acessado em 12/04/2021

Como trabalhos futuros, podemos aprimorar a coleta de dados de modo a unir diferentes bancos de dados e dessa forma construir uma base mais sólida para o treinamento do modelo.

REFERÊNCIAS

ASSUNÇÃO, F. **Estratégias para tratamento de variáveis com dados faltantes durante o desenvolvimento de modelos preditivos**. Tese (Doutorado) — Universidade de São Paulo, 2012.

AZEVEDO, R. B. d. Métodos de machine learning para seleção de variáveis com aplicações ao rugby sevens feminino. 2019.

BATISTA, G. E. d. A. P. et al. **Pré-processamento de dados em aprendizado de máquina supervisionado**. Tese (Doutorado) — Universidade de São Paulo, 2003.

BUDKEWICZ, M. **Aprendizado Supervisionado com exemplos (Supervised Learning)**. 2018. [Online; accessed 12-Abril-2021]. Disponível em: <https://medium.com/horadecodar/aprendizado-supervisonado-com-exemplos-supervised-learning-f9856fed2445>.

GARSON, D. G. 2000. Disponível em: <http//wwz.chass.ncsu.edu/garson/pa765/logistic.htm>.

GOAL, R. **Quais são e quanto pagam os patrocínios máster no Brasil?** 2021. [Online; accessed 12-Abril-2021]. Disponível em: <https://www.goal.com/br/listas/patrocinador-master-times-brasileiros-valores-empresas/jliuotxl769x1grjrvk0q2dcs>.

GONZÁLEZ, L. G. **Análisis de resultados para apuestas deportivas: tenis**. Dissertação (B.S. thesis), 2014.

HONDA, H.; FACURE, M.; YAOHAO, P. **Os Três Tipos de Aprendizado de Máquina**. 2017. [Online; accessed 12-Abril-2021]. Disponível em: <https://lamfo-unb.github.io/2017/07/27/tres-tipos-am/>.

JR, D. W. H.; LEMESHOW, S.; STURDIVANT, R. X. **Applied logistic regression**. [S.l.]: John Wiley & Sons, 2013. v. 398.

MAYRINK, V. Avaliação do algoritmo gradient boosting em aplicações de previsão de carga elétrica a curto prazo. **Technical report**, 2015.

MINUSSI, J. A.; DAMACENA, C.; JR, W. L. N. Um modelo de previsão de solvência utilizando regressão logística. **Revista de Administração Contemporânea**, SciELO Brasil, v. 6, n. 3, p. 109–128, 2002.

MITCHELL, R. **Web scraping with Python: Collecting more data from the modern web**. [S.l.]: "O'Reilly Media, Inc.", 2018.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. **Sistemas inteligentes-Fundamentos e aplicações**, Manole, v. 1, n. 1, p. 32, 2003.

MONARD, M. C.; BARANAUSKAS, J. A. Indução de regras e árvores de decisão. **Sistemas Inteligentes-Fundamentos e Aplicações**, v. 1, p. 115–139, 2003.

NABINGER, A. M. Utilização de algoritmos do tipo machine learning supervisionado para a caracterização dos resultados da copa do mundo de futebol de 2018. 2018.

ONLINE, A. **Apostas esportivas conquistam brasileiros e movimentam R$ 4 bilhões por ano**. 2020. [Online; accessed 12-Abril-2021]. Disponível em: <https://www.apostaonline.com/apostas-esportivas-conquistam-brasileiros-e-movimentam-r-4-bilhoes-por-ano/>.

PADILHA, V. A.; CARVALHO, A. C. P. L. F. Mineração de dados em python. [sn], 2017.

PRATES, W. R. **O que é árvore de decisão (decision tree)? Exemplos em R**. 2018. [Online; accessed 12-Abril-2021]. Disponível em: <https://cienciaenegocios.com/o-que-e-arvore-de-decisao-decision-tree-linguagem-r/>.

RODRIGUES, S. C. A. **Modelo de regressão linear e suas aplicações**. Tese (Doutorado) — Universidade da Beira Interior, 2012.

RON, K.; FOSTER, P. Special issue on applications of machine learning and the knowledge discovery process. **Journal of Machine Learning**, v. 30, p. 271–274, 1998.

SAITO, T.; REHMSMEIER, M. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. **PloS one**, Public Library of Science, v. 10, n. 3, p. e0118432, 2015.

SANCHES, M. K. **Aprendizado de máquina semi-supervisionado: proposta de um algoritmo para rotular exemplos a partir de poucos exemplos rotulados**. Tese (Doutorado) — Universidade de São Paulo, 2003.

SCHMIDT, H. L. Uso de técnicas de aprendizado de máquina no auxílio em previsão de resultados de partidas de futebol. 2017.

SCHNEIDER, C. F. Machine learning aplicado na previsão de resultados de partidas de futebol: um estudo de caso para comparação de diferentes classificadores. 2018.

SILVA, F. R. et al. Uma abordagem para detecção de outliers em dados categoricos. [sn], 2004.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. [S.l.]: MIT press, 2018.