Classificação da Posição de Jogadores do FIFA 23: uma Abordagem de Aprendizado de Máquina

Matheus do Ó Santos Tiburcio[1], **Vinícius Lima da Silva Santos**[1]

[1]Instituto de Computação – Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brasil

{matheusost,viniciuslss}@ic.ufrj.br

Abstract. *This work focuses on applying Machine Learning methods to a dataset containing information about FIFA 23 video game players, aiming to discover the position at which each player performs best based on the provided instances. The study covers data preprocessing, model selection, and visualization of confusion matrices and resulting accuracies, aiming to understand the learning process and challenges encountered during the experiments.*

Resumo. *Este trabalho se concentra na aplicação dos métodos de Aprendizado de Máquina em um dataset com informações sobre os jogadores do videogame FIFA 23, a fim de descobrir a posição que o jogador melhor atua tendo como base as instâncias fornecidas. O estudo aborda desde o tratamento dos dados e escolha dos modelos até a visualização das matrizes de confusão e acurácias resultantes, buscando compreender como se deu o aprendizado e os desafios existentes na realização dos experimentos.*

1. Introdução

O presente estudo visa explorar as técnicas da área de Aprendizado de Máquina, aplicando-as no contexto da classificação das posições dos jogadores de futebol do videogame FIFA 23. Diante dos diferentes modelos de aprendizado existentes na literatura, o trabalho tem como objetivo central descobrir quais destas ferramentas melhor se adequam ao problema em questão e como utilizá-las, de maneira a empregar os principais métodos vistos em aula, pela disciplina de Introdução ao Aprendizado de Máquina, ofertada no Instituto de Computação da UFRJ.

2. Base e Pré-processamento

2.1. Base de Dados

A base de dados utilizada "*Fifa 23 Players Dataset*" conta com as estatísticas oficiais dos jogadores do FIFA 23, um videogame de simulação de futebol publicado pela Electronic Arts, lançado mundialmente em 30 de setembro de 2022 para PC, Nintendo Switch, PlayStation 4, PlayStation 5, Xbox One, Xbox Series X/S e Google Stadia. Os dados foram extraídos do site Kaggle [Kaggle 2010], sendo disponibilizados publicamente em `https://www.kaggle.com/datasets/sanjeetsinghnaik/fifa-23-players-dataset`.

Dentre as informações (atributos) dos jogadores que a base apresenta, este trabalho concentrou-se naquelas voltadas para as habilidades técnicas dos jogadores, haja vista

a necessidade de dados que reflitam as características para a determinação da posição de campo que aquele determinado atleta atua. Nesse sentido, os atributos utilizados como entrada para caracterizar as instâncias foram os atributos ”Crossing”, ”Finishing”, ”Heading Accuracy”, ”Short Passing”, ”Volleys”, ”Dribbling”, ”Curve”, ”Freekick Accuracy”, ”LongPassing”, ”BallControl”, ”Acceleration”, ”Sprint Speed”, ”Agility”, ”Reactions”, ”Balance”, ”Shot Power”, ”Jumping”, ”Stamina”, ”Strength”, ”Long Shots”, ”Aggression”, ”Interceptions”, ”Positioning”, ”Vision”, ”Penalties”, ”Composure”, ”Marking”, ”Standing Tackle”, ”Sliding Tackle”, ”Goalkeeper Diving”, ”Goalkeeper Handling”, ”GoalkeeperKicking”, ”Goalkeeper Positioning”, ”Goalkeeper Reflexes”e ”Preferred Foot”; e o atributo de saída, isto é, aquele que buscaremos prever através dos modelos, foi o atributo ”Best Position”. Vale ressaltar que cada jogador pode jogar em mais de uma posição, tanto no videogame FIFA 23 quanto no mundo real, porém focaremos em determinar, como mencionado, o atributo ”Best Position”, ou seja, a posição que um determinado jogador melhor atuaria dentro do jogo FIFA.

2.2. Pré-processamento dos Dados

Quanto ao pré-processamento realizado, foram feitos os seguintes tratamentos:

1. Remoção das duplicatas
2. Retirada dos outliers
3. Transformação dos atributos categóricos em numéricos (Ordinal Encoding)
4. Junção de todos os atributos de entrada em uma só matriz

Inicialmente, buscamos **remover as duplicatas** de todo o dataset antes de realizar a exclusão dos outliers. Uma vez que cada jogador (linha da base de dados) é único, é possível fazer a **retirada dos outliers** da matriz de atributos numéricos (i.e. todos os atributos de entrada, exceto ”Preferred Foot”). Para os outliers, foi definido que estes seriam os elementos que possuem em, pelo menos um de seus atributos, o Z-Score superior em módulo a 3 desvios padrões. Com os outliers removidos, também foram retirados os elementos referentes a esses outliers da matriz com o atributo categórico de entrada ”Preferred Foot” e da matriz com o atributo categórico de saída ”Best Position”. Posteriormente, foi realizada a **transformação desses atributos categóricos em numéricos** por meio de uma codificação ordinal. Por fim, a matriz contendo os atributo ”Preferred Foot” (já convertido em numérico) **foi integrada à matriz** com os demais atributos de entrada.

Além desse pré-processamento inicial, também foi realizada uma padronização (standardization) das instâncias quando estas foram separadas para o conjunto de treino, o que será melhor abordado na seção seguinte.

3. Definição dos Métodos

3.1. Materiais Usados

Os materiais usados ao longo de todo o estudo, necessários para a aplicação dos métodos e construção dos modelos, foram desenvolvidos tendo como principal ferramenta o *Google Colaboratory* [Google Colab 2017], que armazena os códigos na linguagem *Python* [Python 1991]. Para os códigos, foram utilizadas as bibliotecas *numpy* [NumPy 1995], *pandas* [Pandas 2008], *scikit-learn* [Scikit-learn 2007], *tensorflow* [TensorFlow 2015] e *matplotlib* [Matplotlib 2003].

O relatório de código que contém o notebook do Google Colab utilizado pode ser acessado em: `https://colab.research.google.com/drive/1NeEDQn3P8ykTQAd3l3H9KjHzwwqk_8Y9?usp=sharing`

3.2. Métodos Aplicados no Conjunto de Treinamento

Antes de definirmos os modelos a serem utilizados para a classificação, foram estabelecidos os métodos que seriam aplicados sobre o conjunto de treinamento, sendo esses: o K-Fold Cross Validation como técnica de reamostragem e a padronização (standardization) sobre os valores já separados para treino.

Após o pré-processamento inicial do dataset, os dados foram separados em 70% para o conjunto de treinamento e 30% para teste, de maneira que, com esses 70% para treino, as amostras foram selecionadas com o uso da técnica de K-Fold Cross Validation com $K = 5$, sendo separados novamente grupos para treinamento e validação, dessa vez, próprios do K-Fold.

Com as amostras já selecionadas do K-Fold, realizamos o processo de padronização com o uso das funções `standardScaler.fit` e `standardScaler.transform` da biblioteca *scikit-learn* a fim de que os dados estivessem uniformemente distribuídos segundo uma distribuição normal padrão, ou seja, de média 0 e desvio padrão 1, o que se mostrou relevante antes de submetê-los para treinamento pelos modelos. A partir dos dados padronizados, demos início de fato à construção dos modelos.

3.3. Escolha dos Modelos de Aprendizado

Os modelos escolhidos se baseiam no uso de redes neurais, onde o primeiro modelo trata-se do uso exclusivo de uma rede MLP para realização da classificação dos jogadores, e o segundo utiliza o conjunto de uma rede MLP e um ensemble composto por duas florestas aleatórias (Random Forests) para efetuar essa classificação.

A escolha desses modelos foi feita por meio de uma comparação com os demais modelos vistos em aula. Considerando que o problema abordado se trata de uma classificação a partir de categorias já definidas (i.e. as posições do jogo de futebol), descartamos a utilização de métodos não-supervisionados, tais como a clusterização. Quanto ao uso de regressões, tanto linear quanto logística, e ao uso de árvores de avaliação, foi verifcado empiricamente que esses modelos não se adequavam às instâncias do problema. Já em relação ao uso específico de redes neurais convolucionais, estas também não pareciam uma boa escolha tendo em vista que nosso objetivo não se aproxima do processamento de imagens ou sequer lida com uma alta dimensão de atributos de entrada.

Sendo assim, ao considerarmos o uso de uma rede neural MLP, vimos que a capacidade de fornecer cada informação do jogador como atributo de entrada e testar os neurônios para diferentes camadas de uma rede robusta se mostrou como algo favorável para o uso do modelo. Além disso, quanto ao segundo modelo, a escolha por uma rede neural MLP somada a um emsemble de florestas aleatórias foi feita a partir da visualização do processo de classificação da posição de um jogador como um todo, em que primeiramente seria interessante definir se o jogador é de ataque ou defesa, o que é filtrado pela rede neural, e passar essa classificação mais generalista para uma floresta aleatória especialista, responsável por determinar a posição do jogador de maneira mais específica. Os

testes efetuados que comprovam a acurácia satisfatória desses modelos serão abordados na seção 5. Experimentos Realizados.

4. Trabalhos Relacionados

Dedicaremos esta seção para descrever um dos trabalhos relacionados ao presente estudo. Os trabalho escolhido foi encontrado por meio da busca por *Coding Notebooks* que utilizavam como dataset o mesmo que estamos trabalhando (i.e. o "*Fifa 23 Players Dataset*").

O trabalho, de título "*Linear Regression: Predicting wage of player*" [Dúc Duong 2022], apresenta uma abordagem semelhante ao nosso objetivo, em que, no caso abordado, a motivação é determinar o salário de um jogador (target) com base na nacionalidade dele (feature). Pelo fato do atributo de entrada ser categórico (i.e. os países), foi realizado uma codificação do tipo *OneHotEncoding*, diferentemente da escolhida em nosso caso. Quanto ao modelo de aprendizado, foi utilizada a regressão linear, aplicada por meio da classe `Ridge` da biblioteca *scikit-learn*, em um conjunto separado entre 80% para treinamento e 20% para teste. Os resultados encontrados foram satisfatórios, sendo comparado o salário previsto para o determinado país e a média dos salários daquele país: na Argentina, por exemplo, o salário previsto para um jogador foi de 8022.46 Euros/semana, enquanto a média de salários dos jogadores no país é de 8360.83 Euros/semana.

5. Experimentos Realizados

5.1. Experimentos para o Primeiro Modelo

Para o primeiro modelo, foram realizados os experimentos a fim de verificar a melhor função de ativação e arquitetura de camadas para a rede neural. Foram testadas as funções de ativação *sigmóide*, *tanh* (Tangente Hiperbólica) e *ReLU*. Em todas as arquiteturas, a rede foi construída com 35 neurônios na camada de entrada (cada um referente a um atributo de um jogador) e 15 neurônios na camada de saída (cada um referente a uma posição do futebol). Foram separadas as amostras pelo K-Fold, de modo que, em cada fold, a rede neural foi treinada por 10 épocas. Então, foram realizados os seguintes experimentos de arquiteturas para os neurônios das camadas escondidas, com suas respectivas acurácias médias dos folds, tanto para treino quanto validação, utilizando a função *tanh* (ver relatório de código do Colab para os resultados das demais funções de ativação):

- **1 camada com 1 neurônio**: obteve acurácia média de 0.3383342072367669 para o treinamento e 0.35022683918476105 para a validação.
- **5 camadas com 5, 5, 3, 5 e 5 neurônios, respectivamente**: obteve acurácia média de 0.45760066300630575 para o treinamento e 0.4742284464836121 para a validação.
- **3 camadas com 10, 5, e 10 neurônios, respectivamente**: obteve acurácia média de 0.5809998956322671 para o treinamento e 0.6027620953321456 para a validação.
- **3 camadas com 32, 64, e 32 neurônios, respectivamente**: obteve acurácia média de 0.7176508271694184 para o treinamento e 0.7321264886856078 para a validação.

Diante das acurácias encontradas, escolhemos a *tanh* como função de ativação pelo fato de apresentar uma acurácia média maior que as demais em uma visão geral

dos experimentos. Conforme visto, também optamos pela escolha da arquitetura de 3 camadas escondidas de neurônios com, respectivamente, 32, 64 e 32 neurônios.

Uma vez realizados os experimentos com os conjuntos de treino e validação dos folds, partimos para um experimento com todo o dataset separado entre os 80% de treinamento e 20% de teste propriamente dito. Utilizamos, então, a arquitetura escolhida supracitada, com a função de ativação sendo a *tanh*, e aumentando o treinamento da rede neural para 30 épocas. Assim, o resultado da acurácia na última época obtida foi de 0.7995 para o treinamento e 0.7823 para o teste.

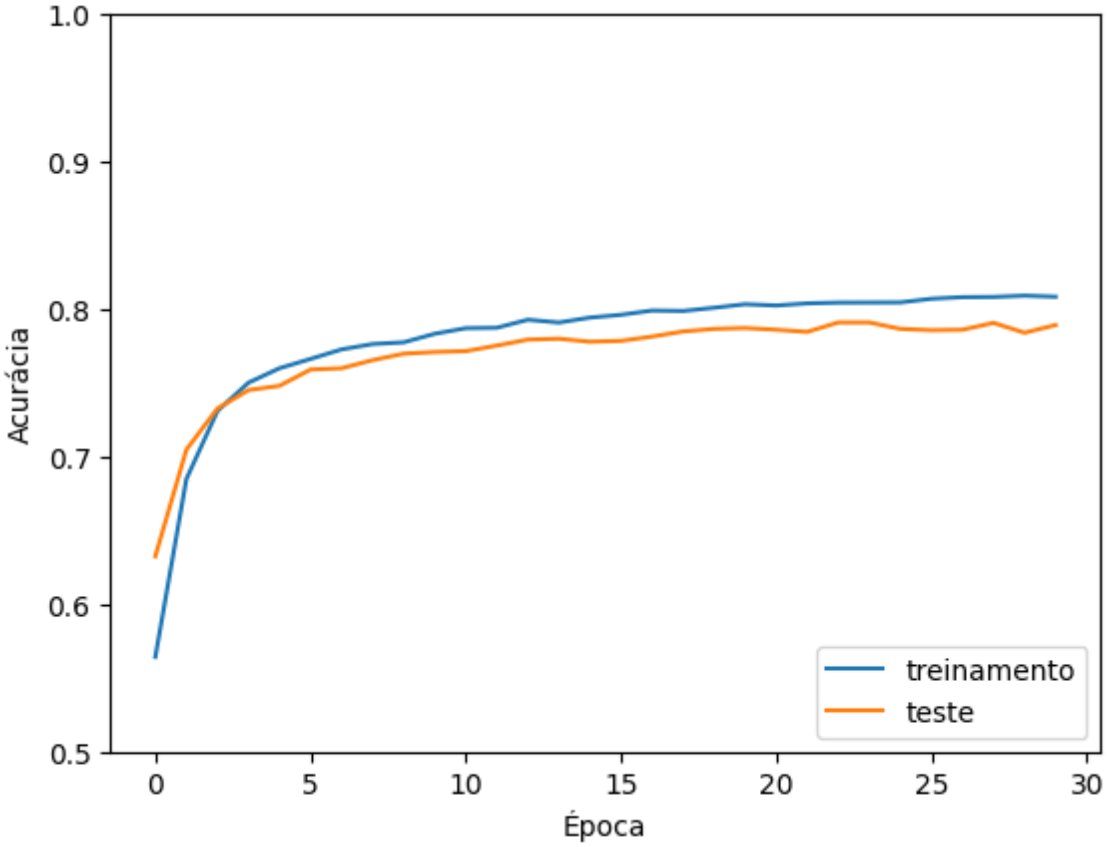

Figura 1. Acurácia por épocas para os conjuntos de treinamento e teste

Também experimentamos, a fins de confirmação, uma arquitetura maior e com mais neurônios, contendo 5 camadas escondidas de 16, 32, 64, 32 e 16 neurônios, respectivamente. Porém, a acurácia encontrada foi muito similar à arquitetura escolhida, o que ratificou nossa decisão pela arquitetura de 3 camadas escondidas de neurônios com, respectivamente, 32, 64 e 32 neurônios, a qual pareceu de fato já estar adequada da melhor forma possível à rede neural do problema.

A seguir, está a matriz de confusão que contém os resultados obtidos no conjunto de teste pela rede neural construída com essa arquitetura, sendo possível visualizar a acurácia das classificações das posições mais detalhadamente:

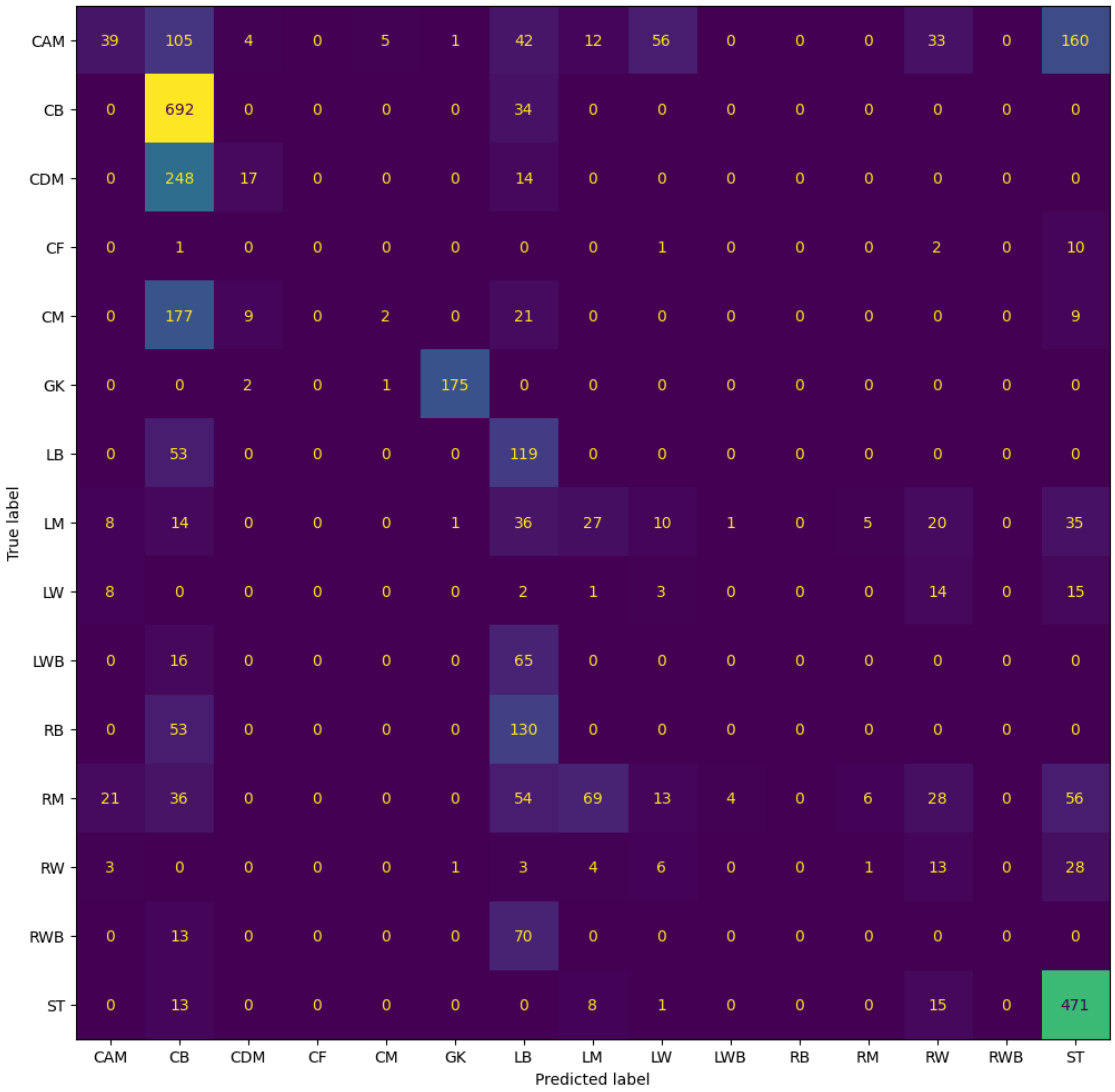

Figura 2. Matriz de Confusão do conjunto de teste para o primeiro modelo

5.2. Experimentos para o Segundo Modelo

Em relação ao segundo modelo, foi feito um experimento para a construção da rede neural responsável pela determinação se o jogador é de uma posição de ataque ou de defesa, em que foi utilizada uma arquitetura com 35 neurônios na camada de entrada (cada um referente a um atributo de um jogador), 2 neurônios na camada de saída (um referente às posições de ataque e outro às de defesa) e uma camada escondida com 1 neurônio, sendo utilizada a *tanh* como função de ativação. Esse experimento foi feito com todas as instâncias do dataset separadas entre 80% de treinamento e 20% de teste, em que a rede neural foi treinada por 10 épocas. Ao verificarmos os resultados, observamos uma acurácia de 0.9464487481117247 para o treino e de 0.9545333957672119 para o teste, o que evidenciou a capacidade desse modelo de rede neural de realizar a categorização entre jogadores de ataque e defesa.

Além dessa rede neural, também foram treinadas as duas florestas aleatórias responsáveis pela classificação da posição mais específica do jogador uma vez que este já foi

categorizado como ataque ou defesa. Ambas os experimentos das florestas foram feitos com todo o dataset também separado entre 80% de treinamento e 20% de teste, sendo utilizadas, em cada uma das florestas, 301 árvores de avaliação. As acurácias obtidas para o teste das florestas aleatórias para os jogadores de ataque e defesa, respectivamente, foram de 0.7514754098360655 e 0.8057291666666667.

Tendo treinado e testado tanto a rede neural quanto as florestas aleatórias, o modelo de forma conjunta foi experimentado, de modo que, primeiramente, foi feita a categorização do jogador como ataque ou defesa pela rede neural para então, posteriormente, ser realizada a classificação da posição mais específica do jogador pela floresta aleatória correspondente. O experimento foi feito com todo o dataset, o qual foi novamente separado entre 80% de treinamento e 20% de teste, obtendo como resultado uma acurácia para o conjunto de teste de 0.663802113575659. É possível visualizar a matriz de confusão encontrada neste experimento a seguir:

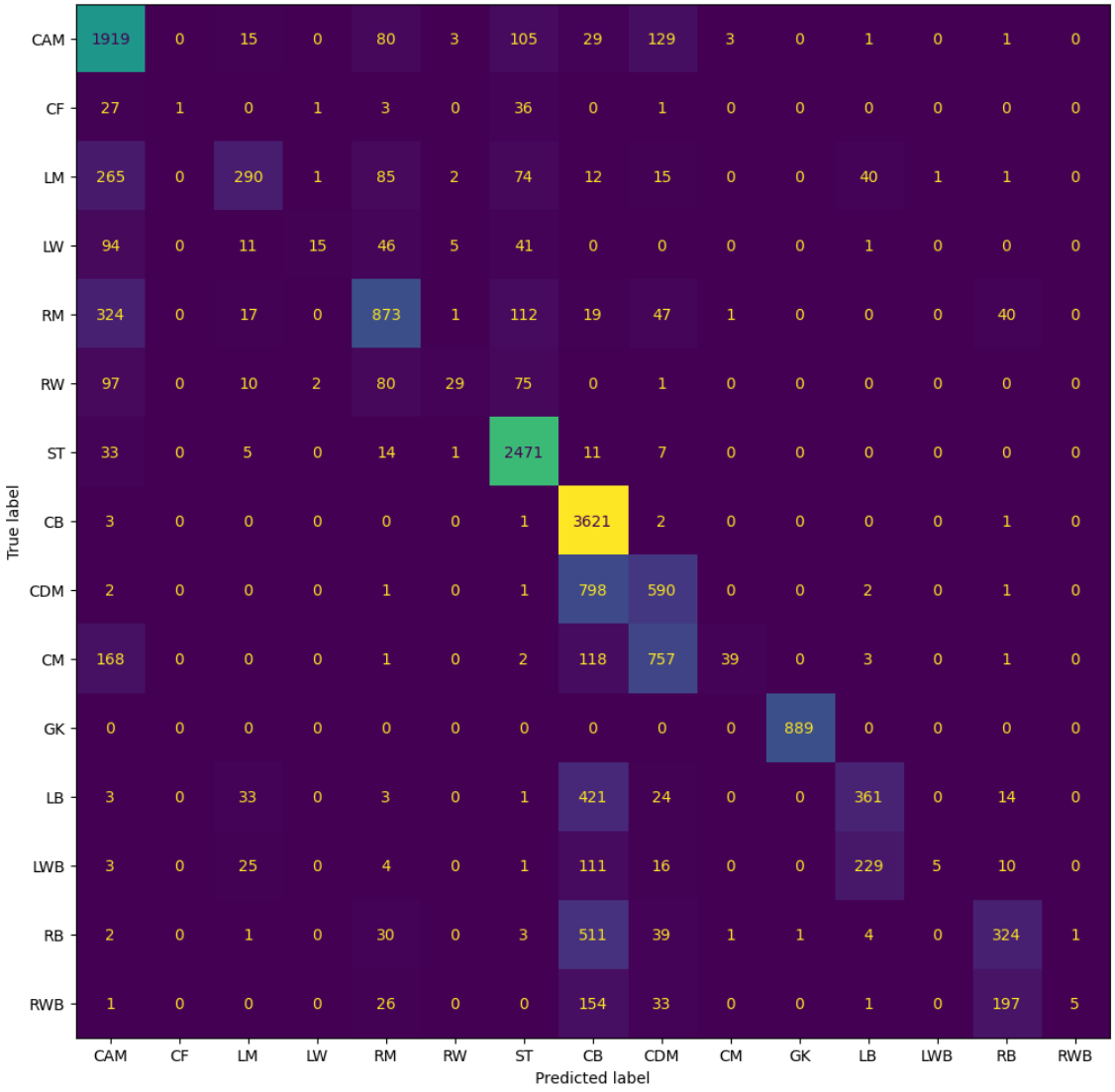

Figura 3. Matriz de Confusão para o segundo modelo

6. Discussão dos Resultados

6.1. Discussão dos Resultados do Primeiro Modelo

Ao analisarmos os resultados obtidos pelos experimentos, constatamos inicialmente, que, de maneira geral, não parecer ter ocorrido overfitting no treinamento do primeiro modelo, isso porque as acurácias médias de treinamento e teste apresentaram valores bem próximos em todos os casos. Nesse sentido, a busca por uma melhora na acurácia desse modelo se deu por meio da análise de como poderíamos mexer de fato em sua estrutura.

Nessa perspectiva, o resultado encontrado pelo experimento das diferentes arquiteturas de rede neural se mostrou eficaz, tendo em vista que a arquitetura final garantiu uma acurácia média de cerca de 80% para o teste com todo o dataset. Pela análise da matriz de confusão gerada, uma ideia de possível melhoria para esse valor de acurácia seria considerar parte das posições do campo que envolvem lados diferentes que o jogador pode jogar como uma só, ou seja, analisar, por exemplo, um meio-campista esquerdo (LM) e um meio-campista direito (RM) como simplesmente uma posição de meio-campista. Muitas das posições incorretamente previstas pela rede neural estão relacionadas a essa dificuldade, o que se deve ao fato das habilidades de jogadores que atuam em posições que se diferenciam apenas nos lados do campo serem muito semelhantes, de modo que o atributo "Preferred Foot" (i.e. se o jogador se considera destro ou canhoto no chute) muitas vezes não ser determinante para garantir que aquele jogador joga pelo lado esquerdo, direito ou central do campo. Além disso, posições do campo muito similares entre si de forma geral também geraram essa confusão pela rede neural, como foi o caso da posição de ala-defensivo direito (RWB) que obteve 35 acertos e 212 erros previstos para a posição de lateral direito (RB), uma função que se aproxima bastante da de ala-defensivo direito no futebol.

6.2. Discussão dos Resultados do Segundo Modelo

Em relação ao segundo modelo, um ponto importante constatado foi o fato da rede neural responsável pela categorização inicial do jogador entre posições de ataque e defesa ter obtido uma acurácia extremamente elevada, de cerca de 96% para o teste, utilizando uma arquitetura com apenas uma camada escondida de 1 neurônio. Esse resultado é interessante justamente porque aparentemente, neste caso, a rede neural agiu como uma espécie de regressão logística, em que o resultado do atributo de saída só possuia duas classificações possíveis (i.e. se o jogador era de uma posição do ataque ou de uma da defesa). Assim, foi possível perceber que, em casos onde se busca prever apenas duas possibilidades de saída a partir de vários valores de entrada, a regressão logística ou a rede neural se mostram indiferentes entre si. Outro fator que atesta a intercambialidade entre os modelos para os diferentes usos é o aplicação bem sucedida de uma regressão linear pelo trabalho de Dúc Duong supracitado, de maneira que, para problemas que recebem apenas um atributo de entrada e retornam a classificação de somente um atributo de saída, a regressão linear, tal como a logística, tal como uma rede neural, seriam modelos eficientes. Portanto, evidencia-se que tudo depende do escopo de variáveis a ser consideradas pelo problema (a rede neural sendo a que suporta um maior número de variáveis, seguida pela regressão logística, seguida pela regressão linear).

Outro ponto analisado no segundo modelo foi uma menor acurácia, de cerca de 66%, quando juntamos a rede neural com as florestas aleatórias. Ao observarmos a matriz de confusão, verificamos o problema do modelo possuir uma "falsa" precisão por

prever corretamente apenas posições que possuem muitas instâncias no dataset, tais como as posições de zagueiro (CB) e atacante (ST). Nesse sentido, grande parte das demais posições de defensores são classificadas erradamente como zagueiro (CB) e, das posições de ataque, como atacante (ST), gerando uma acurácia que apresenta valores altos pelo simples fato de haver mais instâncias dessas posições, e não pelo modelo estar sendo bem sucedido. Dessa forma, quando juntamos a rede neural com as florestas aleatórias, tal problemática se torna mais evidente, e a acurácia do modelo como um todo diminui.

6.3. Aprimoramentos Futuros

Diante dos resultados avaliados, algumas possíveis melhorias a serem implementadas futuramente, a fim de aumentar a acurácia do modelo, são:

- **Balancear o dataset**, o qual atualmente se encontra com muitas instâncias de determinadas classes em detrimento de outras, como é o caso de primeiros atacantes, que contabilizam mais de 3000 instâncias na base de dados, enquanto segundos atacantes contabilizam menos de 100 instâncias.
- **Avaliar a correlação entre os atributos dos jogadores**, uma vez que alguns desses atributos podem ser fortemente correlacionados entre si ou sequer serem correlacionados com o target (i.e. o atributo de saída "Best Position"), o que poderia estar prejudicando a classificação por parte dos modelos.
- **Utilizar um método de clusterização (ou outro não-supervisionado)**, a fim de visualizar que posições, de fato, geram confusão na avaliação dos modelos, tendo em vista que, por mais que empiricamente seja sugestivo acreditar que, por exemplo, um atacante de ponta esquerda possui habilidades muito semelhantes a um atacante de ponta direita, isto não necessariamente se reflete em uma dificuldade para o modelo em si.

7. Conclusão

Conclui-se que as técnicas de Aprendizado de Máquina se constituem como uma ferramenta muito útil para a classificação das posições de jogadores do FIFA 23 com base em informações das habilidades desses jogadores. Não somente essa, mas diversas outras classificações podem ser bem sucedidas com o uso desses modelos de aprendizado. Portanto, evidencia-se a importância pela escolha dos métodos que melhor se aplicam ao problema que se está trabalhando e a necessidade de analisar os resultados por diferentes perspectivas, a fim de obter a melhor acurácia possível para os modelos.

8. Referências

Kaggle (2010). Disponível em `https://www.kaggle.com/`.

Google Colab (2017). Disponível em `https://colab.google/`.

Python (1991). Disponível em `https://www.python.org/`.

NumPy (1995). Disponível em `https://numpy.org/`.

Pandas (2008). Disponível em `https://pandas.pydata.org/`.

Scikit-learn (2007). Disponível em `https://scikit-learn.org/stable/`.

TensorFlow (2015). Disponível em `https://www.tensorflow.org/`.

Matplotlib (2003). Disponível em https://matplotlib.org/.

Dúc Duong (2022). Linear Regression: Predicting wage of player. Disponível em https://www.kaggle.com/code/ducduong18/linear-regression-predicting-wage-of-player.