The Next Football Star:
By: Ben Volovelsky and Izar Hasson
Deep Learning spring 2024

Introduction:

Project goals:

Our Project main goal is to find the next Football stars, We are going to try to achieve that by using data from a few different years. More specifically we will try to estimate a players' potential rating, and to estimate this market value. We will try to do it using deep learning tools learned in class and online. Our goal is to build accurate models that revolutionize how football talent is identified and valued.

Motivation:

Both of us our football enthusiastic and we knew right a ahead that we will want to our project related to that or something similar. We noticed that today football market has an inflation and teams pay ridicules prices on players, sometimes ending as a bust and a lot of money goes down the drain. We are trying to build a tool that can help teams predict the value and upside of a player. Of course the some of this thing are unpredictable and when talking about sports at general and athletes in particular there are surprises. But our motivation is to supply another opinion, a feedback learned from many players and gives an estimation based on years back. The potential rating can give a team a good estimation how far can a player improve years ahead, for the naked eye it seems to be based on current rating, age, and basic football abilities, but our model will search for more sophisticated connections. The market value, pretty straight forwards describes the worth of a player, how much should you pay his team to buy him into your team. It is also based on rating, age, remaining of current contract, and our model will try to find more connections using the features it gets.

Previous work:

We have found during initial research some works exploring ways to estimate a players' price but we wanted to start fresh and didn't base on them. We have added information from a few years back, a potential estimation, and using the FIFA Sony PlayStation game datasets.

Method:

Algorithm:

Let's start with the end, as it is a regression task we will have to decide on a goal, what is "close enough", and how to decide what is a good prediction.

We have explored the data set and the range of potential and value.

Potential is a grade between 1-100, but actually in our dataset between 50-100. We have decided that a 2% error range is acceptable, hence we have tagged a prediction withing a

2% distance from the actual rating as a "hit" and have calculated the total accuracy of the model.

Market value could get a wider range, in our data set between a few thousands and 194 million euros, so we have decided that a 10% mistake range is acceptable.

We have chosen the precent methos cause methods like MSE will create a bias between the highly values players and the low valued once.

Back to the beginning, Firstly we will have to perform Feature Engineering, we have done many elements of features engineering, mainly based on trial and error in order to improve performance:

-Numerical values only: the dataset had many features, some of them were simply irrelevant (players' url, or fifa ID) and some of them were categorical, our model was too complex at the beginning and our dataset was pretty small (will be explained later) so we have decided to limit our model to numerical data only.

-Matching between sets: In order to create a better estimation we have decided to add ratings from 3 different years, they were given to us at 2 different datasets. In order to make no mistakes concatenating the sets we have created a matching function, matching them based on the full name of the player.

-Filling the blanks: The data set had holes in it and we had to decide what to do with it, we have decided to order the players according to their rating and fill the blanks using averages of the adjacent players. This might not be the best estimation of missing data but we didn't have many holes and as the dataset was small we didn't want to remove not full datapoints.

-Scaling: most of the features were simply a grade between 1-100 but some of them weren't (e.g. wage, market value etc.) so we have decided to use a standard scaler in order to prevent a bias. We have also received better results dividing the value in 1 mil making it a price in euro mil instead of in euros.

Architecture:

In our work we present 3 different architectures results, obviously as we are talking about regression tasks, all of the models are built using regressors.

Tabnet: The first model we investigated was Tabnet, as the original dataset is tabular we have wanted to explore the use of tabular transformers. Since we altered the set into a numerical only set results weren't so good as will be presented under the results section.

Stack: Stack is an ensemble learning model which combines multiple machine learning algorithms to improve predictive performance through a technique known as stacking. It integrates four base models: a Multi-Layer Perceptron (MLP) regressor, an XGBoost regressor, a Random Forest regressor, and a Support Vector Regressor (SVR). Each of these models is individually trained on the same dataset, capturing different patterns in the data. Their predictions are then passed to a meta-model, specifically a Ridge regression model, which learns how to best combine these predictions to generate the final output. This approach leverages the strengths of each base model while minimizing their weaknesses, potentially leading to better overall accuracy. On top of that we use cross-validation ensures that the model is robust and generalizes well to new data.

Catboost:

The CatBoost model is a powerful gradient boosting algorithm specifically designed to handle categorical features, although it also excels with numerical data. The model is based on decision trees and includes a depth hyper- parameter to specify the complexity of the individual trees. The model is trained on the training set and validated on the test set during training to monitor performance and prevent overfitting. A combination of high iteration count, deep trees, and careful learning rate adjustment makes CatBoost effective for capturing complex relationships in the data.

When it comes to small datasets, CatBoost's ability to efficiently handle categorical features and its robust regularization techniques make it a strong performer. It is less prone to overfitting compared to other boosting algorithms, thanks to its inherent mechanisms that control over-complexity. Moreover, the model's capacity to work well with small datasets without requiring extensive hyperparameter tuning or feature engineering makes it a practical choice for scenarios where data is limited but accuracy is still crucial. This efficiency, combined with CatBoost's advanced handling of categorical variables, sets it apart as a versatile tool for various machine learning tasks.

Loss function:
For optimization purposes we have used the defaults loss functions of the models: RMSE, but as mentioned above we have design our own accuracy function to estimate our model success. We have tried to use this function as a loss function as well (1-accuracy=loss) but performance wasn't as good.
Experiments and Results:
Dataset:
The datasets provided include the players data for the Career Mode from FIFA 15 to FIFA 22 ("players_22.csv"). The data allows multiple comparisons for the same players across the last 8 version of the videogame.
Each year contains about 10,000 players and 110 columns for each with all kinds of data (numerical and categorical). As mentions above we took only numerical data as it has performed best.
-years selection:
Since we took only players appearing in all the chosen years, we had a tradeoff – taking into account more years back will downsize our dataset. Downsizing the set had a major influence on the accuracy so we have decided to take only 3 yeas data- 2020-2022.
-Use Columns: We had to decide whether to manually filter the features or to let the model do its magic and do it by his own. We have tried to do both and had different results for each prediction (potentials vs market value). Performance was best when we manually filtered for the potentials, and when we didn't filter for the market value, probably because of the wide possible range for market value.

Hyper-Parameters tuning: In order to tune the hyper-parameters model we have used Optuna. Optuna is a tool learned in class. It gets as an input a range to test for each hyper-parameter and performs an algorithm to find the best hyper-parameters. Unfortunately, running optuna took a lot of time as we couldn't use it for as many trials that we wanted

with our resources (time and computational) so we have used it for 10 runs and used it's best results as our model hyper-parameters.

We have seen right away that our best performing model is catboost se we have focused our efforts on it and used the other only as baseline models and the best hyper-parameters received from optuna were on trial 2 out of 10 and they are: 4711 iterations, depth tree of 10, and learning rate of 0.01.
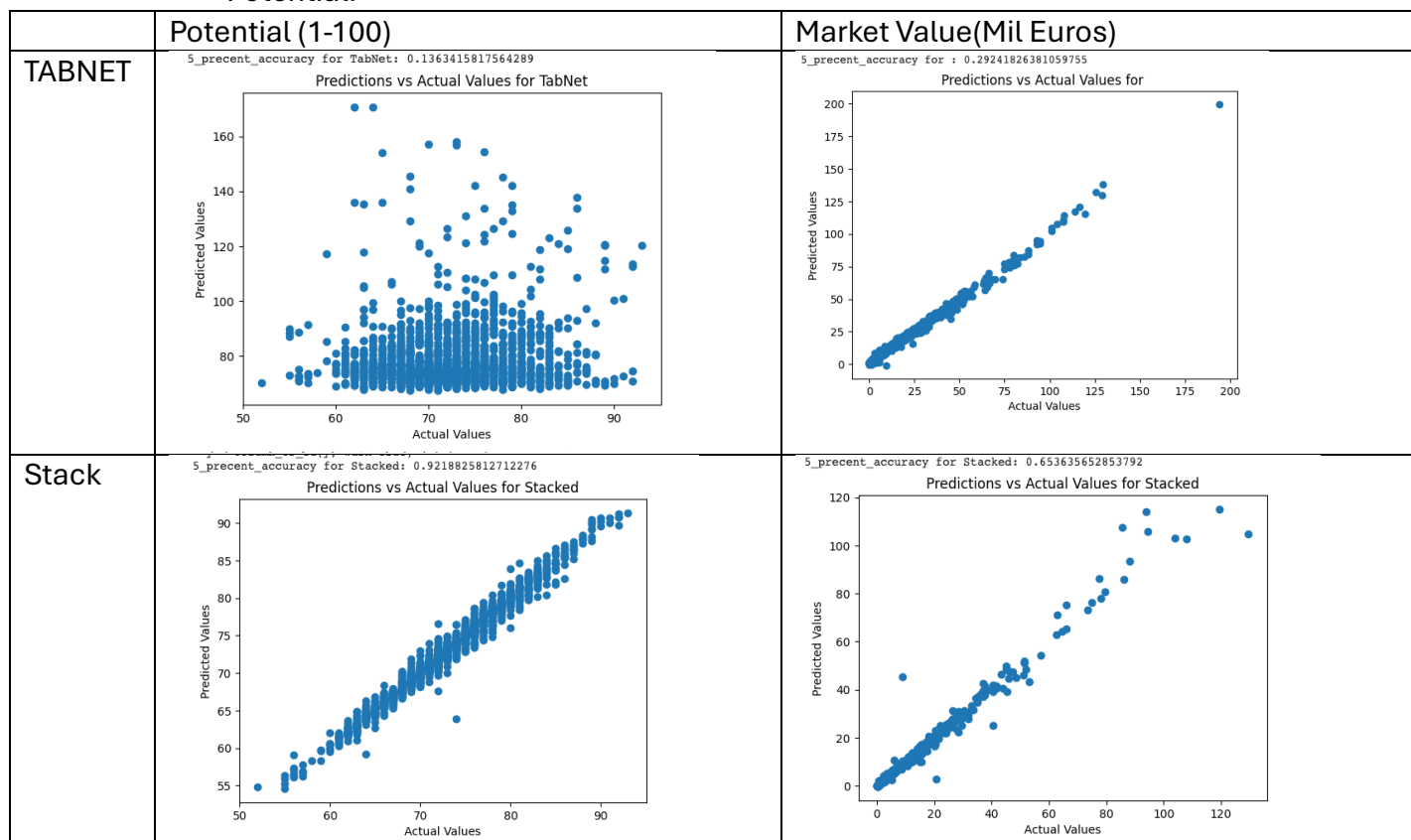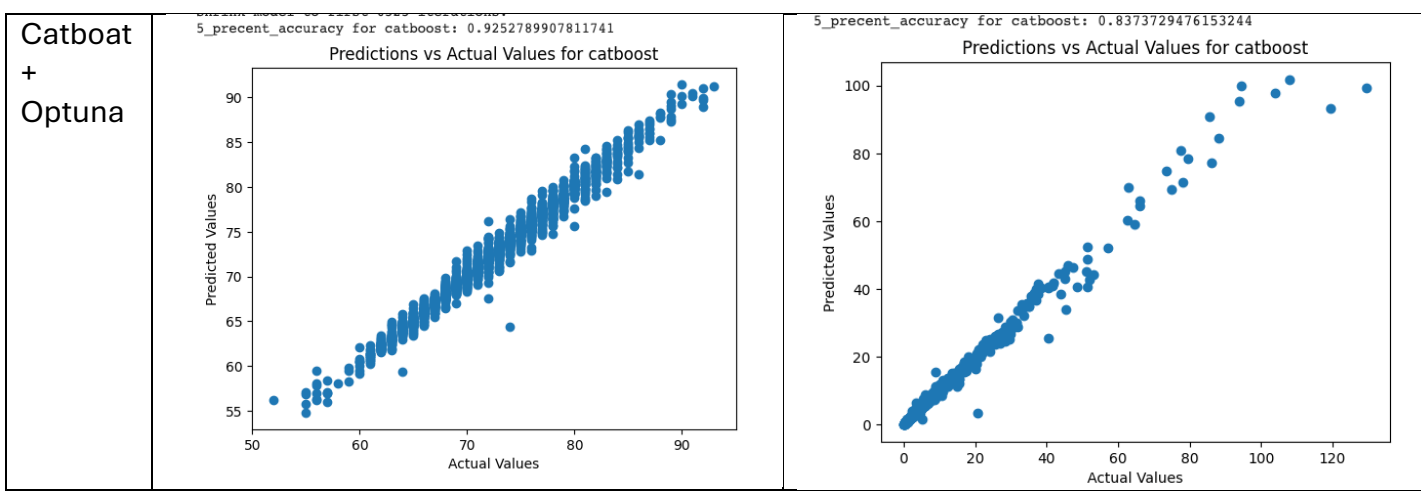
Evaluation:

We have used 2 main metrics to estimate our model success:

1) The mentioned above "custom accuracy" which gets a mistake range (10% for market value and 2% for potential) and calculated accuracy according to whether the prediction was within error range or not

2) A graph plotting the actual value vs the predicted value. We expect to see a straight diagonal line (y=x) for a good performing model as each predicted value will match its actual value exactly).

Results:

In the following figure we can see the results for each model for both Market Value and Potential:

	Potential (1-100)	Market Value(Mil Euros)
TABNET	5_precent_accuracy for TabNet: 0.1363415817564289	5_precent_accuracy for : 0.29241826381059755
Stack	5_precent_accuracy for Stacked: 0.9218825812712276	5_precent_accuracy for Stacked: 0.653635652853792

Catboat + Optuna	 5_precent_accuracy for catboost: 0.9252789907811741 Predictions vs Actual Values for catboost	 5_precent_accuracy for catboost: 0.8373729476153244 Predictions vs Actual Values for catboost

1) Tabnet Performance: we can see that Tabnet performed really bad on the set, perhaps further tuning and modifications would have improve it's performance but we can see that it wasn't even close. For potential predictions it even values over a 100 which is not possible, and it didn't even plot a diagonal nice line like the others, so we have looked for alternatives.
2) Stack have actually performed pretty well, on potential it is even close to catboost, but we weren't happy with its accuracy over the market value and we once again looked for an alternative.
3) Catboat has straight away improved our accuracy, in the plot below you can see how it has already performed well before using Optuna, and how hyper-parameters tuning with Optuna have improved results dramatically:

	Without Optuna	With Optuna
Catboat for player market value	 Shrink model to first 4562 iterations. 5_precent_accuracy for catboost: 0.7311984473556526 Predictions vs Actual Values for catboost	 5_precent_accuracy for catboost: 0.8373729476153244 Predictions vs Actual Values for catboost
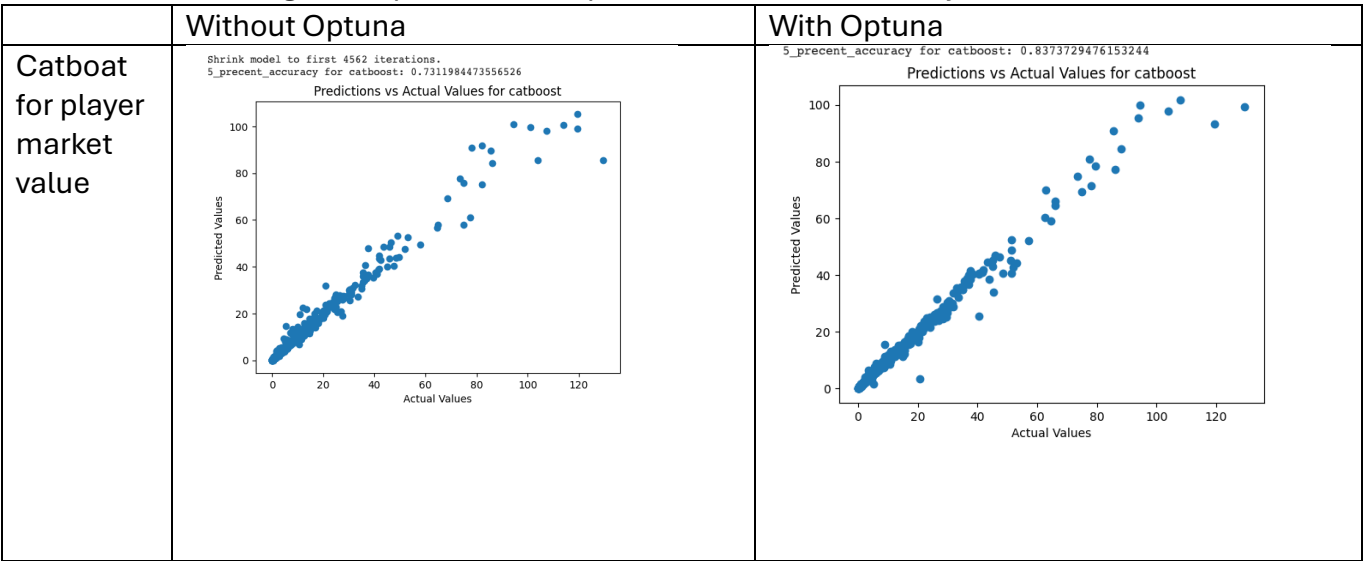
Pure Deep Learning:
Our last part of the research will compare pure deep learning elemetns (as boosting doesn't fall under the category of deep learning). We have decided to explore 3 different pure deep learning models, and to use the on with the best reults fo further training.

The 3 models that we have studied our:

CAN regressor: Causal Attention Networks are a specialized architecture in deep learning designed to model and infer causal relationships within data, particularly in sequence-based tasks like time series analysis or natural language processing. Unlike traditional attention mechanisms, which focus on learning correlations by weighing different parts of the input data, CAN explicitly incorporates causal inference principles to identify and emphasize the cause-effect relationships between variables. This allows the network to not only make accurate predictions but also to understand the underlying causality within the data. By combining the strengths of attention mechanisms with causal reasoning, CAN enhances the model's interpretability and robustness, particularly in complex systems where understanding the cause-effect dynamics is crucial.

CNN regressor: Convolutional Neural Network Regressor is a type of neural network designed for tasks that involve predicting continuous values, leveraging the power of CNNs. Traditionally used in image processing, CNNs are known for their ability to capture spatial hierarchies and local patterns through convolutional layers. In the context of regression, CNN Regressors apply this capability to predict numerical outputs, such as age estimation from facial images, price prediction from housing features, or any other task where the input data has a grid-like structure, such as images or time series. The architecture typically involves a series of convolutional and pooling layers to extract relevant features, followed by fully connected layers that map these features to a continuous output. This makes CNN Regressors particularly effective in tasks where the spatial or temporal structure of the input data is crucial for accurate prediction.

MLP: Multy Layer Preceptron is a neural network model, we have built using PyTorch a specifically designed NN for regression tasks. It features a customizable architecture with multiple hidden layers, each defined by a specified number of neurons. The class incorporates key elements for enhancing performance and regularization: LeakyReLU activations are used to introduce non-linearity while mitigating the risk of dying neurons; Batch Normalization is applied to stabilize and accelerate training by normalizing layer inputs; and Dropout is included to prevent overfitting by randomly dropping a fraction of neurons during training. The final layer outputs a single value, making this model well-suited for continuous value prediction tasks. This modular design allows for flexibility in adjusting the architecture to fit different input dimensions and task requirements.

In order to find the best MLP architecture we have used optuna and trial and error until we found the sufficient architecture.

Loss Function: We have detected that as there is a big different between the range of player value it is more well suited to use the L1 loss function. The MSE have added a bias due the big difference and using the L1 function have benefitted the results.

Feature Selection: After trial and error, we have discovered that the results were dramatically better using only the "use-columns"- a filtered dataset using only columns that have seemed to us as relevant rather using the entire raw data.

Epochs: Training the 3 models was very time consuming, we haver decided to train each one for 500 epochs and to choose the best one for further training, as the harder task is predicting the player market value, we have examined the 3 models on it and we present the results in this table:
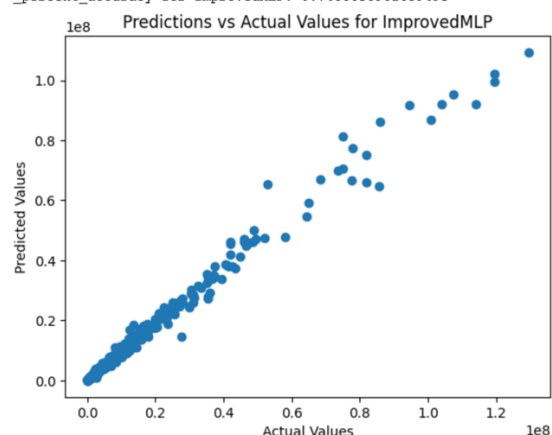
	Market Value(Mil Euros)
CAN	_percent_accuracy for CAN: 0.3279961183891315
CNN	_percent_accuracy for CNNRegressor: 0.23629306162057254
MLP	_percent_accuracy for ImprovedMLP: 0.651140223192625

We have chosen to proceed with the MLP model, we have trained it for 3500 epochs this time and received these results which are sufficient results for both potential and market value:
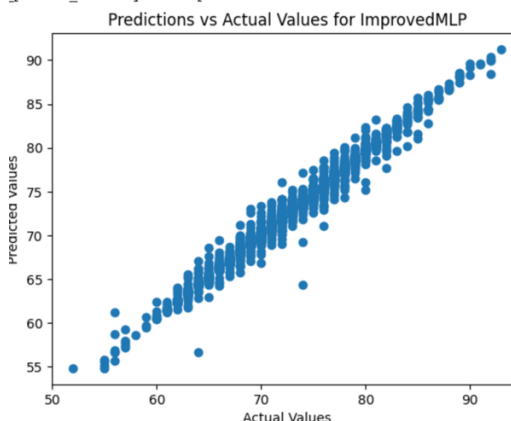
	Market Value (EUR)	Potential (1-100)

MLP 3500 epochs	_percent_accuracy for ImprovedMLP: 0.7486656962639495 Predictions vs Actual Values for ImprovedMLP	_percent_accuracy for ImprovedMLP: 0.9121785540999515 Predictions vs Actual Values for ImprovedMLP

Conclusions:

-Tabular regressors don't work well on numerical data. Results of the Tabnet model were very bad and we had to look for alternatives, loading up models into an ensemble have actually improves resulted dramatically, but at the end of the day, the simple solution of catboosting have yielded the best results but it isn't a deep learning solution.

-Hyper-parameters are important: We have used Optuna as a final step improving our best model- catboost. It ran slowly but the hyper-parameters it supplied have improved our results dramatically – over 10% accuracy improvement

-Feature engineering had a major effect on our model selection and our results, we have started working on the whole dataset and repeated trial and error have led us to the formula that supplied us with the results that we have desired. We have learned that there are trade-offs building a set which integrates a few different sets, and that different predictions demands different modifications to the model: features engineering, and hyper-parameters as well (We didn't change architecture).

-Scaling had a major impact on our results. Although we have used standard scaler on the whole set, dividing the value of a player in a million improves our model performance.

-Boosting worked out great for our model, we have learned online about boosting gradient and how it works and have seen that it improves our model dramatically.

-MLP have yielded the best results out of the "pure" deep learning models. It out performed CAN and CNN dramatically at the first 500 epochs and showed great results under 3500 epochs. Finding the best architecture for the MLP wasn't easy but after trial and error and fine tuning we have found great results.

Ethics Statement:

1)Introduction:

Student names: Ben Volovelsky and Izar Hasson

Project Title: The next Football Star

Project Description: Our model aims to find the next football starts using data from a few different years, documented in the FIFA Sony PlayStation game. It aim to estimate a players' potential and Market Value using deep learning tools.
2)We have used ChatGPT as the LLM model to answer our questions:

2) a. List 3 Types of Stakeholders

1. Football Clubs and Scouts: These are the primary users of the model, as they will leverage it to identify emerging talent and make informed decisions on player acquisitions.
2. Game Developers (e.g., EA Sports): The developers who create the FIFA video game will be stakeholders since the model uses their data, and the model's findings could influence future game development or updates.
3. Players and Agents: Football players and their representatives will be affected by the predictions related to a player's potential and market value, which can impact contract negotiations and career planning.

b. Explanation to Each Stakeholder

- Football Clubs and Scouts: "Our model uses historical data from the FIFA PlayStation game to predict which young players are likely to become future stars, helping you make more informed decisions on player acquisitions. By analyzing patterns in player development and market value, the model can highlight promising talents who may be under the radar, offering a competitive edge in player recruitment."
- Game Developers: "Our project utilizes data from the FIFA Sony PlayStation game to build a model that predicts football players' potential and market value. This model not only demonstrates the value of the data in real-world applications but also provides insights that could be used to enhance future iterations of the game, particularly in career mode features that focus on player development."
- Players and Agents: "The model we've developed analyzes FIFA game data to estimate a player's future potential and market value. This can offer players and their agents a new perspective on their career trajectory, helping to set realistic goals and make strategic decisions regarding transfers and contract negotiations based on predicted future performance."

c. Responsibility for Giving the Explanation

- Football Clubs and Scouts: The responsibility of communicating with football clubs and scouts would likely fall to the Data Science Team Lead or Product Manager. They would need to ensure the model's capabilities and benefits are clearly understood, as well as provide actionable insights that these stakeholders can use in their scouting and recruitment processes.

- Game Developers: The Project Manager or Business Development Lead would be responsible for explaining the project to game developers. They would focus on highlighting how the model's use of the game's data demonstrates its value beyond entertainment, potentially influencing future game development or partnerships.
- Players and Agents: A Sports Analyst or Public Relations Manager would be best suited to communicate with players and their agents. This person would need to frame the model's predictions in a way that is empowering rather than threatening, emphasizing how the data can assist in career development and decision-making.

3) Ethnical Biases- as we have seen in class, AI models might include ethnic bias, in the FIFA based next star model, decisions are based on ethnic and physiological features such as weight, nationality, height etc. Players might fear from fatphobia or geographical bias and therefore explanations to every stakeholders must include detailed step being actively done to prevent this kind of biases.