Improving the Expected Goal value in football using Multilayer Perceptron Networks*

Manuel Méndez, Carlos Montero, and Manuel Núñez[0000−0001−9808−6401]

Design and Testing of Reliable Systems research group
Universidad Complutense de Madrid, Madrid, Spain
manumend@ucm.es,cmonte09@ucm.es,manuelnu@ucm.es

Abstract. The development of big data and machine learning is having a special boost in the field of innovation in sports. Football is the most popular sport in Europe with millions of players and billions of euros invested. Currently, machine learning applications in football are focused on video analysis to events detection (tracking of players, statistics matches, scouting, ...), injuries evaluation and prediction, among others. The xG metric (Expected Goals) determines the probability that a shot will result in a goal and is often displayed on screen during the most important football matches in the world. In this paper we present a new model to obtain more accurate predictions of the probability of a shot becoming a goal. The model is based on a multi-layer perceptron neural network (MLP), which allows the interaction of different variables and improves the model's performance. Our proposal includes an evaluation of the quality and a comparison with the xG provided by statsbomb, one of the most important football data providers of the world. The results show that our model outperforms the quality of the expectations provided by statsbomb. Specifically, it is clearly better in their capability to detect actual positive cases (goals).

Keywords: Deep Learning in Sports; Big Data in Football; Expected Goal; Multilayer Perceptron Networks

1 Introduction

The widespread popularity of big data and machine learning [19, 23, 31] in recent years has led to their extended use in various and heterogeneous fields [12, 13, 18, 20]. One area in which they have driven significant innovation is sports, particularly in football. Currently, it is common for top football teams to employ specialists who analyse a multitude of data variables to identify areas for improvement in upcoming games. Big data is used in several areas of football [3, 11, 28], including player performance analysis, tactical analysis, injury prevention, and scouting. In player performance analysis, big data helps analysts to identify the strengths and weaknesses of individual players, allowing coaches to make informed decisions about including them in the team. In tactical analysis, big data is used to detect patterns in opponents game-play, enabling coaches

* This work has been supported by the Spanish MINECO/FEDER project AwESOMe (PID2021-122215NB-C31) and the Region of Madrid project FORTE-CM (S2018/TCS-4314), co-funded by EIE Funds of the European Union.

Threshold	Balanced accuracy	F1-score	Recall	Precision
0.2	0.741	0.517	0.584	0.463
0.5	0.655	0.446	0.333	0.677
0.75	0.617	0.37	0.251	0.7
0.9	0.5	0	0	0

Table 1. xG by metric and threshold above which a shot is considered goal in World Cup 2022.

to develop effective strategies to counter them [5]. Big data can also be used to monitor player fitness levels and detect early signs of injury, helping teams take proactive measures to prevent injuries [17]. Finally, big data is used in scouting, allowing teams to identify potential transfer targets based on their performance statistics in other leagues and countries [2].

One of the most popular metrics used to analyse the performance of a football player is the expected goals (xG) generated. This metric aims to determine the probability (between 0 and 1) that a shot will result in a goal. In recent years, the popularity of this metric has increased considerably, to the point that it is now often displayed on-screen during the most important football matches in the world when a shot occurs. In current models, xG is obtained from a logistic regression [24] based on historical records that models the probability that a goal occurs after a shot. This model uses as independent variables the distance to goal, angle to goal, body part of the shot, type of assist, goalkeeper position, position of all attackers and defenders and shot impact height.

However, after an analysis of the quality of the xG model most frequently used[1], we detected some shortcomings. For example, when considering the World Cup 2022 games, we noted that only 6.42% of shots had an xG higher than 0.5. This percentage was reduced to 2.23% if we discard penalty shots. However, the number of goals was 10.63% when discarding penalties and 13.6% when including them. Taking these statistics into account, we concluded that, in general, the xG assigned to any given shot is too low. We continued to analyse the xG in the World Cup 2022 using classical statistical metrics and obtained poor results in most cases, as we can see in Table 1, using different thresholds above which we predict the shot to be a goal (by default, this value is 0.5).

After reviewing the available scientific literature, we realised that deep learning has not been yet used to predict xG; only pure statistical models have been used. Therefore, we decided to explore new alternatives and models to obtain more accurate predictions of the probability of a shot becoming a goal. We believe that one of the reasons for the poor performance of xG models is that while each variable has a determined weight, the relationship between two or more variables is not explored by the model, which is a significant limitation in evaluating a shot. Therefore, we propose a model based on a multi-layer perceptron neural network (MLP), which allows the interaction of different

[1] Not all xG models take into account the same factors. We refer in this work to the xG model given by Statsbomb [25], which uses more contextual events and better quality data than any other provider to accurately measure the quality of chances.

variables and improves the model's performance. Our proposed model is trained with data obtained from *statsbomb* [25], which contains many characteristics of more than $16,000$ shots from 19 top-level football competitions played between 2005 and 2022. In order to evaluate the quality of our model, We used five classical statistical metrics for binary classification and a new metric developed by us that directly compares models.

The rest of the paper is organised as follows. In Section 2 we review the most relevant and recent contributions related to the topics of the paper. In Section 3 we present the evaluation metrics that we will use to compare the quality of the results provided by our proposal with the ones provided by statsbomb. In Section 4 we describe the data extraction and preprocessing processes and give the main characteristics of our proposal model, including the hyperparameters fine-tuning. In Section 5 we present the results of our experiments. Finally, in Section 6 we present our conclusions and some lines for future work.

2 Related work

xG is a well-known metric that has been used for several years to predict the number of goals in a game. Initially, this metric was calculated based on factors such as goals scored in previous games, average number of goals, average number of goals playing at home or away, first or second half performance, etc. For many years, the Poisson distribution has been the most commonly used model for this purpose. The method was first used to simulate the World Cup 1998 [4]. Over the years, this method has been improved, and it was further refined in 2010 to simulate the World Cup in South Africa [30]. The next major improvement was in the World Cup 2014 [9] where various potentially influential covariates were included. It is important to note that the model predicted the actual world champion, Germany. Other international competitions where the Poisson Distribution (and its improvements) has been employed include the UEFA European Championship 2016 [8], where a bivariate Poisson distribution was used, and the African Championship 2019 [7], where authors employed a nested regression Poisson model.

In recent years, with the significant development of machine learning and real-time technology, the xG metric is no longer calculated solely based on previous goal variables, but also takes into account shots taken during the game. Each shot is assigned a probability between 0 and 1 of being a goal. The prediction of xG has been studied less in the scientific community, mainly due to the difficulty of obtaining data with a high number of shots and enough characteristics of each shot to apply the models. The logistic regression model is the most commonly used model in this case [24]. Although there are not many scientific papers about it, it is widely known that this model is used in the xG metric that appears in the most important football data websites of the world [6, 29], as well as in Statsbomb [25], the site from where we obtain data and use it as a benchmark to compare our proposal.

We would like to mention several studies that have used either statistical machine learning techniques or deep learning models in football video analysis. An extensive survey [1] has reviewed the challenges (e.g. player/ball detection and tracking, event detection, and game analysis) and compared the different deep learning-based meth-

ods and their performance. There are methods for the extraction of player's trajectory in a video football game [10]. In this case, the deep learning technology is used for automatic extraction of the features of the players, detection and tracking.

Machine learning methods have been recently used to work with trajectories of the players in the pitch. By comparing actual movements with the reference movements generated via trajectory prediction, it is possible to predict the likelihood of a player without the ball to create a scoring chance for teammates [27]. Another model shows that the first five seconds that a team has possession are fundamental to determine whether the team will be able to create a goal chance [26].

Finally, another line of work focuses on the efficient fatigue monitoring and in maintaining the best performance of players. A deep learning algorithm has been used to predict the Rate of Perceived Exertion (RPE) from players' movement [14]. The model preprocessed the raw GPS data to obtain linear and angular components of velocity, acceleration, and jerk routes. Another proposal uses a multi-dimensional approach to injury forecasting in professional soccer that is based on GPS measurements and machine learning, using GPS tracking technology, and then construct an injury forecaster [22].

3 Evaluation metrics

In this section, we will describe discuss the performance evaluation metrics, that is, a number of metrics employed to evaluate the quality of a binary classification model.

Let TP be the number of true positives, TN be the number of true negatives, FP be the number of false positives, FN the number of false negatives, and $T = TN + TP+FN+FP$ be the total number of observations. We consider the following metrics to evaluate classification models. *Accuracy* (Acc). This is the most typical metric for a classification model. The accuracy measures the proportion of correct predictions made by the model over the total number of predictions. The mathematical formulation of the accuracy is:

$$Acc = \frac{TN + TP}{T}$$

Although accuracy is a commonly used metric for evaluating classification models, it can be misleading in unbalanced problems. When the distribution of classes in the data is imbalanced, accuracy can be biased towards the majority class, leading to misleading results.

Balanced Accuracy (B_Acc). This metric addresses the limitations of Accuracy by taking into account the class imbalance in the dataset. It gives equal weight to both classes, regardless of their number of observations, and provides a more accurate measure of the model performance in classifying both classes. The mathematical formulation of the balanced accuracy is:

$$B\_Acc = \frac{\frac{TP}{TP+FN} + \frac{TN}{TN+FP}}{2}$$

Precision ($Prec$). This metric indicates the model ability to accurately identify the positive cases. The mathematical formulation of the precision is:

$$Prec = \frac{TP}{TP + FP}$$

```python
def c_met(model1,model2,obs):
    aux1 = []
    aux2 = []
    for i in range(len(obs)):
        if obs[i] == 1:
            if model1[i] > model2[i]:
                aux1.append(True)
            else:
                aux2.append(True)
        if obs[i] == 0:
            if model1[i] < model2[i]:
                aux1.append(True)
            else:
                aux2.append(True)
    if sum(aux1) > sum(aux2):
        print('Model 1 is the best')
    else:
        print('Model 2 is the best')
```

Fig. 1. C_metr implementation in Python.

Recall (Rec). This metric indicates the quantity of positive cases that the model can identify. The mathematical formulation of the precision is:

$$Rec = \frac{TP}{TP + FN}$$

F1-score ($F1$). This metric combines recall and precision in only one value. The mathematical formulation of the F1-score is:

$$2 \cdot \frac{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}}{\frac{TP}{TP+FP} + \frac{TP}{TP+FN}} = 2 \cdot \frac{Prec \cdot Rec}{Prec + Rec}$$

Comparison metric (C_metr). We developed a metric to facilitate the direct comparison of two models in a binary classification context. The metric operates by assigning a point to the model that assigns a higher probability to an observation that is truly positive and to the model that assigns a lower probability to an observation that is truly negative. Following this process for all observations, the model with the highest number of points is deemed the superior performer. The implementation of this metric in Python is shown in Figure 1

4 Data preprocessing, problem description and proposed model

In this section, we will describe the process of data extraction and preprocessing and the main characteristics of the problem at hand, including the definition of our model. Finally, we present the fine-tuning process applied to obtain the best performance of our model.

4.1 Data acquisition, preprocessing and predictor variables

Our data was obtained from `https://github.com/statsbomb/open-data` and downloaded using the *statsbomb* package in Python. The dataset includes a large

number of top football games, along with their corresponding events such as shots, passes, and fouls. In this paper, we focus only on shots. We extracted shots performed in nineteen competitions, including the World Cup (2018 and 2022) and the Spanish First Division (from the 2004-2005 to 2020-2021 seasons), resulting in a total of $16,042$ shots. Each shot is associated with 33 variables. Next, we will explain why we decided to discard sixteen of these variables and focus solely on the remaining seventeen for our prediction. We discarded four variables related to the identification of the shot. We also discarded the date, possession, possession team, team name, and player name. Then, we discarded whether the shot was off-camera, as well as the outcome of the shot (because this is the variable we will predict) and whether it was the first shot of the game. We also discarded the variables related to the coordinates that refer to the end location of the shot because our prediction will be given based only on the beginning of the shot, not the end. Finally, we discarded the expected goal according to *statsbomb* because this is the variable against which we will compare our model results.

Therefore, we will use the remaining seventeen variables, which are: *period*, *minute*, and *second*, the *play pattern* (i.e., a regular play, a shot from a free kick, a shot from a throw-in, etc.), the *position* of the player who takes the shot, the *duration* of the ball possession by the player who takes the shot before the play, whether the player who takes the shot is *under pressure* or not, the *body part* that is used to take the shot, the *type* of shot, the *technique* used to take the shot (volley, half-volley, header, back-heel, etc.), whether the shot *is followed by a drive* or not, whether the shot is *redirected* or not, whether the play is a *one-on-one* situation or not, whether the shot is taken on an *open goal* or not, whether the shot is *deflected* or not, and, finally, the coordinates x and y from which the shot is taken. Obviously, the *outcome* of the shot (whether the shot is or not a goal) is the target variable.

Once we have identified the predictor variables and the target variable, we need to describe how we obtained accurate, consistent, and complete data to work with, a process known as *data preprocessing*. First, we take the categorical variables, including the target variable, (they are all except x, y, *second*, *minute*, and *duration*), and apply a technique called *LabelEncoder*. This technique is commonly used to transform a categorical variable into a numerical variable. Then, we apply a *StandardScaler* method to all the variables (except to the continuous, that is just 1 or 0). This method is used to transform numerical data to have a mean of 0 and a standard deviation of 1. It works by subtracting the mean of each variable and then dividing the result by the standard deviation of that variable. Finally, the last step of the preprocessing phase is to ensure that the target variable is properly coded, assigning a value of 1 to the cases where a goal has been scored and a value of 0 to the cases where a goal has not been scored.

In conclusion, our objective is to predict, with an error as small as possible, whether a shot is a goal (y) by using the 17 predictor variables ($X_1, \ldots, X_{17}$) that we described in the previous lines.

4.2 Proposed model

Our proposal model is a Multi-Layer Perceptron Neural Network (MLP). This is a type of artificial neural network that consists of multiple layers of interconnected nodes, or neurons, which are organised into input, output, and hidden layers. Each neuron

Hyperparameter	Set of values
h	$\{1, 2, 3, 4, 5\}$
h_i	$\{16, 32, 64, 128, 256\}$
lr	$\{0.001, 0.01, 0.1\}$
dr	$\{0.2, 0.3, 0.4\}$
ep	$\{10, 20, 30\}$
bs	$\{32, 64, 128\}$

Table 2. Grid of hyperparameters considered in the fine-tuning.

receives inputs from the previous layer and produces an output that is passed on to the next layer. The neurons in the hidden layers use an activation function to transform the input signals and produce a nonlinear output. The output layer produces the final output of the network, which is typically used to make predictions or classify input data. MLPs are trained using a supervised learning algorithm, such as backpropagation, to adjust the weights and biases of the neurons in the network to minimise the error between the predicted output and the actual output.

The operations occurring in every neuron in the hidden and output layers can be represented by the following equations:

$$H_x = f(b_1 + W_1 \cdot x) \qquad O_x = f'(b_2 + H_x \cdot W_2)$$

being x an input vector, b_1 and b_2 bias vectors, W_1 and W_2 weight matrices and f and f' activation functions. Usual activation functions are the RELU and sigmoid functions.

$$RELU(a) = \max(0, a) \qquad Sig(a) = \frac{1}{1 + e^{-a}},$$

where a is the input data.

Hyperparameters such as the *number of hidden layers* (h), *number of neurons in hidden layer* i (h_i), *learning rate* (lr), *dropout rate* (dr), *number of epochs* (ep), and batch size (bs) will be fine-tuned in order to obtain the optimal performance.

4.3 Hyperparameters fine-tuning

In order to perform fine-tuning, we use GridSearch, which is a hyperparameter tuning technique that systematically searches through a specified hyperparameter space to find the optimal combination of hyperparameters that result in the best performance of the model. In GridSearch, a grid of hyperparameter values is defined, and the model is trained and evaluated for each combination of hyperparameters in the grid. The hyperparameters that yield the best performance, as measured by a chosen evaluation metric, are then selected as the optimal hyperparameters for the model. It is usual to improve the quality of the fine-tuning process by combining it with the cross-validation technique. In this case, we develop the grid of hyperparameters shown in Table 2.

We chose this hyperparameter grid, with a low number of layers and epochs, because our dataset has a small number of independent variables. We also used prototypical numbers for the neurons (powers of 2), learning rate (negative powers of 10), epochs (multiples of 10), and batch size (also powers of 2).

δ	Acc		B_Acc		$Prec$		Rec		$F1$	
Model	Proposal	*sts* model	Proposal	*sts* model	Proposal	*sts* model	Proposal	*sts* model	Proposal	*sts* model
0.25	0.868	0.871	0.695	0.707	0.479	0.466	0.463	0.48	0.479	0.492
0.5	0.875	0.871	0.697	0.75	0.443	0.273	0.47	0.79	0.456	0.407
0.75	0.874	0.893	0.712	0.781	0.403	0.174	0.496	0.798	0.445	0.29

Table 3. Metrics comparing our proposal vs. *statsbomb* model.

We used the *Balanced Accuracy* evaluation metric to determine the best performance, as this metric gives equal weight to both categories and does not discriminate between them. Additionally, we employed a cross-validation method with 3 folds, meaning we split the data into 3 different testing and training sets to prevent over-fitting.

The optimal values obtained according to the best performance in *Balanced Accuracy* are: $h = 4$, $h_1 = 256$, $h_2 = h_3 = h_4 = 64$, $lr = 0.1$, $dr = 0.2$, $ep = 20$ and $bs = 32$. These results show that higher values of the hyperparameters cannot be associated with better performance, as might be expected in cases such as the number of layers, neurons, batch size, or epochs.

5 Experiments

In this section, we compare our (fine-tuned) model with the xG metric provided by *statsbomb*, taking into account all the metrics described in Section 3. To conduct the experiments, we randomly divided the dataset into a training set and a testing set with a ratio of $9/10$ and $1/10$, respectively. We then applied our model with the optimal hyperparameters obtained from the fine-tuning process described in the previous section. After running the experiments, we compared and analysed the evaluation metrics applied to the results of our model and the xG metric provided by *statsbomb*.

In order to compare our proposed model and the xG values provided by *statsbomb*, we need to set a threshold. If a model outputs a value above this threshold, δ, for an observation, then the observation will be predicted as a goal. On the contrary, if a model outputs a value below the threshold, then the observation will be predicted as no goal. By default (and logically), the most obvious value of δ is 0.5. However, depending on the context, this value can be higher or lower. In some models, it may be preferred to maximise sensitivity (proportion of true negatives) instead of specificity (proportion of true positives). In other models, depending on the cost of error, it may be more appropriate to increase or decrease the threshold. In order to explore all the options, we will perform the comparison setting different values of δ.

Table 3 shows the evaluation of the performance of our model and the *statsbomb* model according to five metrics and three different thresholds. We can draw several conclusions. Firstly, we can observe that our model performs better than the model provided by *statsbomb* in some cases, while for others the *statsbomb* model is better. Note that when focusing on accuracy and balanced accuracy, both models perform similarly, but our proposed model is slightly worse than the *statsbomb* model regardless of the threshold. However, these metrics do not provide a complete understanding of the model's quality in an unbalanced classification problem and we are working with

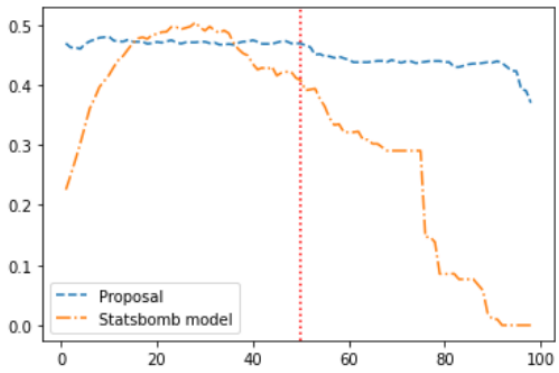

Fig. 2. $F1$ (y axis) by δ value (x axis) in the proposal model and in the model provided by Statsbomb.

a clearly unbalanced one. To gain a better understanding, we need to examine metrics that assess the models' ability to classify negative (no goal) and positive (goal) values. Therefore, we use Precision and Recall, which show very different behaviours of the models. While a better precision denotes a better capability to detect the proportion of true positives among all the positive predicted cases, and is useful when minimising false positives is important, a better recall denotes the capability to detect the proportion of true positives among all the positive cases and is useful when minimising false negatives is important.

The results show that our model clearly outperforms the *statsbomb* model in precision, while the opposite occurs in recall. Because of this, we appeal to F1-Score, a metric that combines both Precision and Recall. By using this metric, the results vary depending on the threshold: with $\delta = 0.25$, the *statsbomb* model is slightly better than our proposal, with $\delta = 0.5$, our proposal is slightly better than the *statsbomb* model, and with $\delta = 0.75$, our proposal clearly outperforms the *statsbomb* model.

Figure 2 illustrates the F1-score as a function of the δ value, ranging from 0 to 0.99. We observe that the F1-score in our model remains constant, while it does not in the model provided by *statsbomb*. Our proposed model outperforms the one provided by *statsbomb* for all δ values except for the range between 0.2 and 0.4, where the F1-score of the *statsbomb* model is slightly higher than our proposal. For the remaining values, our proposal clearly outperforms the *statsbomb* model, including $\delta = 0.5$ (red dotted line), which is the "logical" value that the threshold should receive.

Overall, when using metrics such as Accuracy and Balanced Accuracy, which do not provide a good understanding of unbalanced classification tasks, our proposal performs slightly worse than the model provided by Statsbomb. However, when using specific metrics, our proposal obtains better precision, while the *statsbomb* model obtains better recall. This indicates that our proposal is better at minimising false positive cases (which is a major issue in most xG models), while the *statsbomb* model is better at minimising false negatives. When unifying both metrics, our proposal obtains better results, as can be seen with the F1-Score metric.

		Actual	
		Negative	Positive
Pred	Negative	1314	107
	Positive	90	94

		Actual	
		Negative	Positive
Pred	Negative	1390	146
	Positive	14	55

Table 4. Confusion matrix ($\delta = 0.5$) of our model (left) and of the Statsbomb model (right).

In order to provide a more comprehensive understanding of what metrics mean, we will present and discuss the confusion matrices, see Table 4, obtained using a threshold of $\delta = 0.5$. The confusion matrices illustrate the main difference between the models, which was also reflected in the metrics: our proposed model detects more true positives (i.e. goals) at the expense of an increase in false negatives, while the model provided by Statsbomb can only detect a low number of true positives, resulting in an increase in false positives.

Finally, we evaluated the C_{metr} metric as defined in Section 3. Our proposed model clearly outperformed the model provided by Statsbomb in 82.5% of the cases, indicating that our model predicted the actual outcome (i.e. goal, represented by 1 or no goal, represented by 0) more accurately.

6 Conclusions and future work

In the field of professional football, there is an increasing demand for technical analysis of the teams and players performance. The Expected Goal (xG) is a usual indicator of that performance, which measures the probability of a shot becoming goal. In this paper, we have developed a new method based on a multi-layer perceptron neural network (MLP) to obtain more accurate predictions of the expected goal. The model uses as independent variables the distance to goal, angle to goal, body part of the shot, type of assist, goalkeeper position, position of all attackers and defenders and shot impact height. This work allows the interaction of different variables and improves the model's performance, specially by detecting a higher number of positive cases.

Focusing on the results, we compared our proposal with the *statsbomb* model using several metrics. We addressed some questions, such as the threshold, δ, which represents the probability above which a shot is considered a goal. The accuracy and balanced accuracy metrics do not provide a good understanding of unbalanced classification tasks. Both models obtained similar results regardless of the δ value used. However, the recall metric showed that the *statsbomb* model performed better, while the precision metric indicated that our proposal performed better. We then used the F1-Score metric, which combines precision and recall, to compare the models. Our proposal outperformed the *statsbomb* model according to F1-score. Finally, according to the C_metr metric, our proposed model outperformed the model provided by Statsbomb in 82.5% of the cases, indicating that our MLP model more accurately predicted the actual outcome.

We consider some lines of future work. First, since the field is growing, we plan to review the work on machine learning applied to sports. We will follow a method-

ology similar to the one used in our recent work [16]. Second, due to the high quality and different perspective of the proposed model, we will apply it to more data from professional football leagues to minimise errors by training with a high number of observations (we need to improve/increase the data obtained from *statsbomb*). Third, in addition to the goal, other football plays such as shooting and saving penalties may also be determinants in the match, and similar models to ours can be applied in these cases. Fourth, we rely on a supervised model but it would be interesting to consider an unsupervised approach following recent work on attacks detection [21]. Finally, we would like to implement a model similar to the one described in the introduction that predicts the xG of a team in a match, rather than of a shot. In this case, the model would be based on the team's historical xG in previous games, rather than their historical goals, and might be useful to consider a hybrid approach as we did in our recent work [15].

References

1. Sara Akan and Songül Varli. Use of deep learning in soccer videos analysis: survey. *Multimedia Systems (in press)*, 2023.
2. Iman Behravan and Seyed Mohammad Razavi. A novel machine learning method for estimating football players' value in the transfer market. *Soft Computing*, 25(3):2499–2511, 2021.
3. Silvia Cacho-Elizondo and José-Domingo Lázaro Álvarez. Big data in the decision-making processes of football teams. *Journal of Strategic Innovation and Sustainability*, 15(2), 2020.
4. D. Dyte and S. R. Clarke. A ratings based poisson model for world cup soccer simulation. *Journal of the Operational Research Society*, 51(8):993–998, 2000.
5. Lei Fang, Qiang Wei, and Cheng Xu. Technical and tactical command decision algorithm of football matches based on big data and neural network. *Scientific Programming*, 2021:1–9, 04 2021.
6. Footystats. `https://footystats.org/es/spain/la-liga/xg`. Accessed: 2023-03-10.
7. Lorenz A. Gilch. Prediction Model for the Africa Cup of Nations 2019 via Nested Poisson Regression. *African Journal of Applied Statistics*, 6(1):599 – 616, 2019.
8. Andreas Groll, Thomas Kneib, Andreas Mayr, and Gunther Schauberger. On the dependency of soccer scores - a sparse bivariate poisson model for the uefa european football championship 2016. *Journal of Quantitative Analysis in Sports*, 14(2):65–79, 2018.
9. Andreas Groll, Gunther Schauberger, and Gerhard Tutz. Prediction of major international soccer tournaments based on team-specific regularized Poisson regression: An application to the FIFA World Cup 2014. *Journal of Quantitative Analysis in Sports*, 11(2):97–115, 2015.
10. Xin He. Application of deep learning in video target tracking of soccer players. *Soft Computing*, 20(20):10971–10979, 2022.
11. Tim A. Herberger and Christoph Litke. The impact of big data and sports analytics on professional football: A systematic literature review. In Tim A. Herberger and Jörg J. Dötsch, editors, *Digitalization, Digital Transformation and Sustainability in the Global Economy*, Springer Proceedings in Business and Economics, pages 141–171. Springer, 2021.
12. Eklas Hossain, Imtiaj Khan, Fuad Un-Noor, Sarder Shazali Sikander, and Md Samiul Haque Sunny. Application of big data and machine learning in smart grid, and associated security concerns: A review. *Ieee Access*, 7:13960–13988, 2019.
13. Ryan H. L. Ip, Li-Minn Ang, Kah Phooi Seng, J. C. Broster, and J. E. Pratley. Big data and machine learning for crop protection. *Computers and Electronics in Agriculture*, 151:376–383, 2018.

14. Jeongbin Kim, Hyunsung Kim, Jonghyun Lee, Jaechan Lee, Jinsung Yoon, and Sang-Ki Ko. A deep learning approach for fatigue prediction in sports using GPS data and rate of perceived exertion. *IEEE Access*, 10:103056–103064, 2022.
15. M. Méndez, M. G. Merayo, and M. Núñez. Long-term traffic flow forecasting using a hybrid CNN-BiLSTM model. *Engineering Applications of Artificial Intelligence*, 121:106041, 2023.
16. M. Méndez, M. G. Merayo, and M. Núñez. Machine learning algorithms to forecast air quality: a survey. *Artificial Intelligence Review (in press)*, 2023.
17. Ting Meng, Jing Yong Yang, and De Yi Huang. Intervention of football players' training effect based on machine learning. In *2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE'22)*, pages 592–595, 2022.
18. Kee Yuan Ngiam and Wei Khor. Big data and machine learning algorithms for health-care delivery. *The Lancet Oncology*, 20(5):e262–e273, 2019.
19. Junfei Qiu, Qihui Wu, Guoru Ding, Yuhua Xu, and Shuo Feng. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, 2016:1–16, 2016.
20. Jose F. Rodrigues, Larisa Florea, Maria C. F. de Oliveira, Dermot Diamond, and Osvaldo N. Oliveira. Big data and machine learning for materials science. *Discover Materials*, 1:1–27, 2021.
21. J. Roldán-Gómez, J. Martínez del Rincón, J. Boubeta-Puig, and J. L. Martínez. An automatic unsupervised complex event processing rules generation architecture for real-time IoT attacks detection. *Wireless Networks (in press)*, 2023.
22. Alessio Rossi, Luca Pappalardo, Paolo Cintia, F. Marcello Iaia, Javier Fernández, and Daniel Medina. Effective injury forecasting in soccer with GPS training data and machine learning. *PLOS ONE*, 13:1–15, 07 2018.
23. Surender Reddy Salkuti. A survey of big data and machine learning. *International Journal of Electrical & Computer Engineering*, 10(1), 2020.
24. Soccermatics. fitting the xg model. `https://soccermatics.readthedocs.io/en/latest/gallery/lesson2/plot_xGModelFit.html`. Accessed: 2023-03-10.
25. Statsbomb. `https://statsbomb.com`. Accessed: 2023-03-10.
26. Leandro Stival, Allan Pinto, Felipe dos Santos Pinto de Andrade, Paulo Roberto Pereira Santiago, Henrik Biermann, Ricardo da Silva Torres, and Ulisses Dias. Using machine learning pipeline to predict entry into the attack zone in football. *PLOS ONE*, 18:1–24, 01 2023.
27. Masakiyo Teranishi, Kazushi Tsutsui, Kazuya Takeda, and Keisuke Fujii. Evaluation of creating scoring opportunities for teammates in soccer via trajectory prediction. In *9th International Workshop on Machine Learning and Data Mining for Sports Analytics (MLSA'22)*, pages 53–73. Springer, 2023.
28. Poojan Thakkar and Manan Shah. An assessment of football through the lens of data science. *Annals of Data Science*, 8:823–836, 2021.
29. Understat. `https://understat.com/`. Accessed: 2023-03-10.
30. Dingwei Wang. Soccer tournament simulation and analysis for south africa world cup with poisson model of goal probability. In *2010 Chinese Control and Decision Conference*, pages 3654–3659, 2010.
31. Lina Zhou, Shimei Pan, Jianwu Wang, and Athanasios V Vasilakos. Machine learning on big data: Opportunities and challenges. *Neurocomputing*, 237:350–361, 2017.