Assigning Field Positions to Football Players

Pritish Gulati
IIIT, Delhi
pritish19076@iiitd.ac.in

Pulkit Piplani
IIIT, Delhi
pulkit19080@iiitd.ac.in

Shayna Malik
IIIT, Delhi
shayna19206@iiitd.ac.in

Varun Khurana
IIIT, Delhi
varun19124@iiitd.ac.in

November 25, 2021

Abstract

Determining the best player position for a player is a key to the success of a team in soccer. The purpose of this research is to find out the best playing position of a player, by assessing the attributes of each player from the FIFA dataset. We propose the use of the FIFA Soccer video game dataset [1], as there is a lack of a large scale dataset with player attributes. To find out the best position of each player, we have implemented several models such as MultiClass-Logistic Regression, Naive Bayes, Decision Trees and Random Forests.

1 Introduction[1]

Football (soccer) is the most viewed sport in the world. The ability of a player to perform in a football match is largely influenced by the position the player plays in. There have been several instances where players playing out of position start to underperform, for example-: Griezmann playing at RW instead of CF in Barcelona.

Putting together a successful football team depends on the manager's ability to determine the best suitable position for a player in a team and the correct formation. In order to aid the manager, our system makes use of certain attributes such as speed, skills, movement, power and defensive attributes to find out the best position for a particular player.

[1]`https://github.com/pritish19076/AssigningPlayerPositionsUsingML`

2 Literature Survey

In [2], Abraham E. Evwiekpaefe et al. (2020) used Artificial Neural Network to classify soccer players into 3 categories - 'Good', 'Average', and 'below average' with respect to their ability to play in any Forward Position. They selected data of 100 players from the FIFA-2017/18 complete dataset. They considered various attributes such as Acceleration, Agility, Balance, Ball control etc to classify the player where each attribute was categorized as 'Good', 'Average' and 'Below Average'. The ANN model outperformed the J48 classifier with an accuracy of 0.68.

Leonardo Cotta et al (2016) [3] studied the evolution of German and Brazilian soccer players throughout the years and the unique playing style of FC Barcelona. They achieved this by calculating the mean of the attributes of all players of the respective from the FIFA video game dataset. Linear Regression was used to study the evolution and they concluded that the Germans had evolved much more offensively. K-Means clustering was used to cluster popular football teams ,which led to a conclusion that more than a unique style, FC Barcelona had unique midfielders.

3 Dataset

3.1 Exploratory Data Analysis

We are using FIFA-21 complete player dataset [1]. It contains data of the players of last 7 years of various soccer leagues. It contains 106 columns of which we are using 52 columns. For reducing the complexity of our problem, we have categorized all player positions on the basis of their horizontal position on a football field namely, ‘L’ - Left, ‘C’ - Center and ‘R’ - Right and then on the basis of vertical position on the field namely, ‘B’ - Back, ‘M’ - Middle and ‘F’ - Forward and also the special position GK - Goalkeeper. Overall, this results in 10 separate classes for the player positions.

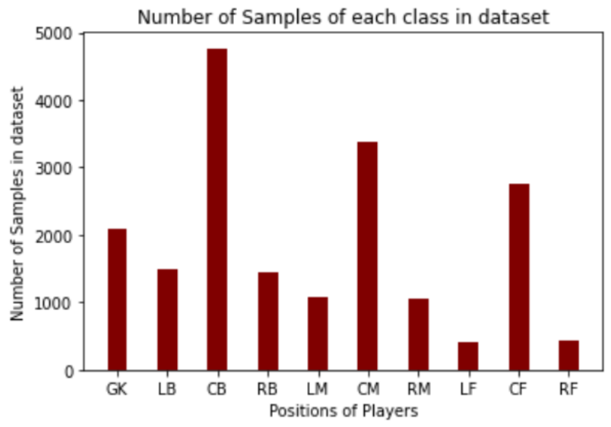

Figure 1: Number of samples for each class

By comparing number of samples of each class, it is evident that ’LF’ and ’RF’ are the minority classes and ’CB’ and ’CM’ are the majority classes. Thus, to handle class imbalance, we have used **CTGAN**[7]. ’CTGAN is a collection of Deep Learning based Synthetic Data Generators for single table data, which are able to learn from real data and generate synthetic clones with high fidelity’.

Figure-2 shows that, more left footed players play on the left side and more right footed players play on the right side. Also, we infer that a large majority of players playing in the centre are right footed. Similarly, Figure-3 shows that forwards and midfielders have maximum movement stats followed by defenders and goalkeepers.

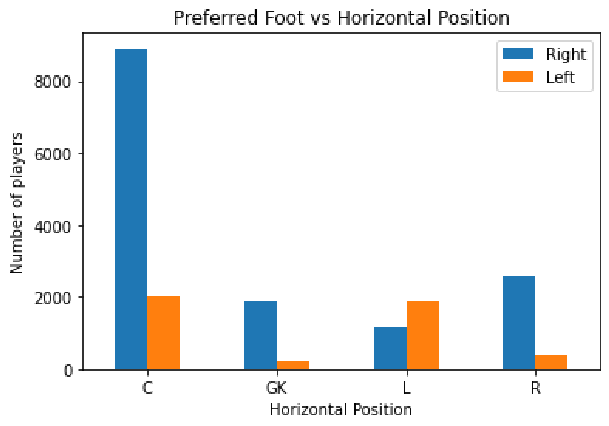

Figure 2: Distribution of Players across Horizontal positions and their preferred foot

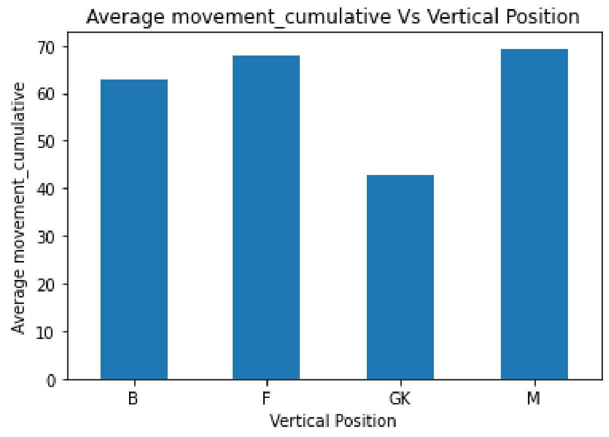

Figure 3: Distribution of Players across Vertical positions with their movement attributes

Visualisation using PCA

PCA is used to reduce the number of dimensions of data while trying to retain maximum information stored in it. It tries to maximize the amount of variation retained from the original data distribution. We use PCA to reduce the dimensions of the data points to three dimensions and plot the same. Through this we can visualise higher dimensional data and get a sense of similarity between data points.

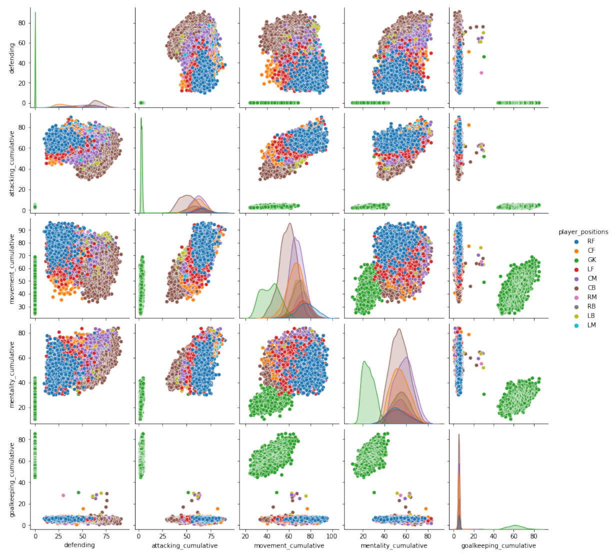

Figure 4: Pairplots of selective features

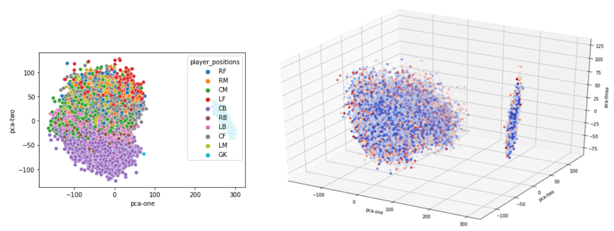

Figure 5: PCA plots - 2D and 3D

3.2 Dataset Preprocessing

Some features which were not helpful in determining the positions were dropped such as 'name', 'club name', 'jersey number' etc. Features that are closely related such as 'skill_dribbling', 'skill_curve' etc were clubbed into one cumulative feature with name - attribute_cumulative. To calculate the cumulative attribute, mean of similar columns was taken and put into one column. For categorical data, we have used one hot encoding. We replaced NAN values by 0 as cells which were not filled was not valid for that particular player or their position. Used Gaussian Standardization for Feature scaling as most of the features closely followed Gaussian distribution. Figure-6 provides the motivation for Gaussian Standardization. Mapped the soccer player positions to 10 positions by clustering closely related positions. Figure-7 depicts the mapping.

Even after above preprocessing, some features such as pace, shooting, attacking_cumulative were highly correlated to each other. To reduce it, we again clubbed some of the relevant features such as 'pace', 'shooting', 'passing' and 'dribbling' were clubbed with 'attacking_cumulative'. This step reduced the number of features as well as correlation between other features as well.

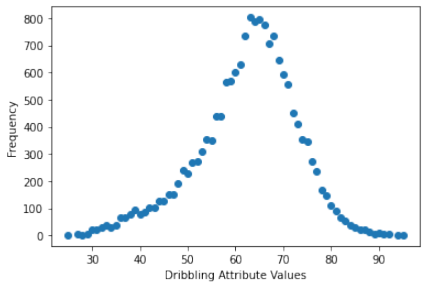

Figure 6: Distribution of players wrt. their Dribbling attribute values

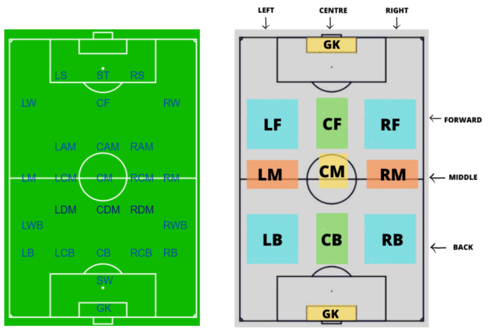

Figure 7: Position Mapping

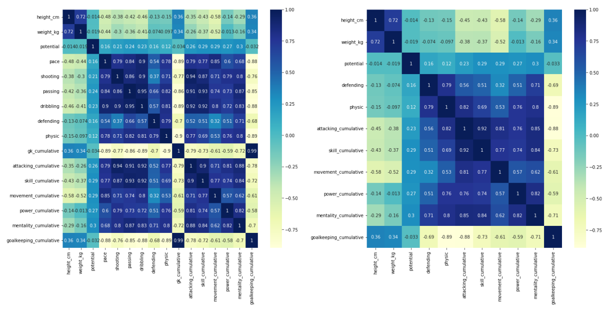

Figure 8: Correlation Matrix for Features

4 Methodology

We have used the following models to predict the best position for a player given attributes :

1. **Multiclass Logistic Regression**: Used in prediction for more than 2 classes. It is used when the dependant variable is categorical and when the data is case-specific.

2. **Random forest**: It is an ensemble learning method as it uses many decision trees and the ouput is taken based upon the majority vote and due to this, it performs better.

3. **Decision Tree**: It is used mainly for classification. It partitions the dataset in a nonparametric fashion.

4. **Gaussian Naive Bayes**: It is used for continuous data. It is based on bayes theorem and is simple yet has high functionality.

5. **SVM**: Support Vector Macine is mainly used for classification and regression problems. It is beneficial in case of high dimensional spaces. It can solve both linear as well as non linear problems.

6. **MLP**: Multilayer perceptron is an artificial neural network. It is mainly used for classification and prediction problems. It consists of a number of nodes connected to each other just like in the case of a directed graph.

7. **ADABoost**: It combines weak classifiers(decision stumps) to create a strong classifier. It keeps on running until a state wherein no further improvement is acheived or it has created a fixed number of weak classifiers. It works upon the concept of up-weighing observations which were incorrectly classified previously.

8. **XGBoost**(Extreme Gradient Boosting) : It controls overfitting by using regularized model formalization. While reducing the cost, it gives more weightage to functional space rather than to hyperparameters.

9. **Gradient Boost**: Mainly useful in case of unbalanced data. It is more flexible as compared to ADABoost. It is based on minimising the loss function.

10. **K Nearest Neighbours**: The K-NN algorithm assumes the similarity between the new case/data and available cases and puts the new case into the category that is most similar to the available categories.

We propose two methods to predict the position which are as follows:

1. Train a multiclass classifier to predict the position of player ({ GK, LB, CB, RB, LM, CM, RM, LF, CF, RF }) on which the player is expected to perform the best.

2. Train two models, one will predict the best vertical position ({ GK, B, M, F }) and the other one will predict the best horizontal position ({ L, C, R }) for the player. We combine the outputs of both the models and predict the final position for the player.

For all the models, we have implemented the following pipeline:

- Data preprocessing (as shown in Section 3.2).
- Split the dataset into train and test sets.
- Run grid search with cross validation to find best possible hyperparameters (if any).
- Fit the model using best hyperparameters found. Predict position for players in the test set.
- Evaluate the model on various metrics (Accuracy, Precision, Recall, F1-Score).
- Perform 5-Fold cross validation to evaluate performance of the model.

Model	Method-1	Method-2
Multiclass Logistic Regression	0.746	0.71
Gaussian Naive Bayes	0.644	0.557
Random Forest	0.736	0.731
Decision Tree	0.654	-

Table 1: Test Set Accuracy of models

5 Results and Analysis

Method-1 performs better than Method-2 as the first method penalises the classification model in case it makes a wrong prediction where as in method-2 one of the model might not be penalised for it's wrong prediction and hence combining the outputs of the two models reduces the accuracy. Thus, we analyse rest of the models using method-1 only.

Model	Hyperparameters
Multiclass Logistic Regression	(solver='newton-cg', penalty='None')
Gaussian Naive Bayes	-
Decision Trees	(criterion='gini', max_depth=10)
Random Forest	(max_depth=19, criterion='gini', max_features='sqrt')
ADA Boost	(n$_{e}stimators = 140$)
Gradient Boost	(n$_{e}stimators = 130$)
XG Boost	(n$_{e}stimators = 70$)
Multi-layered Perceptron	(activation='relu', hidden$_{l}ayer_{s}izes = [10, 5], lr =' invscaling', lr_{i}nit = 0.001$)
Support Vector Machine	(kernel='rbf', degree=1)
K Nearest Neighbours	(weights='distance', n$_{n}eighbours = 21$)

Table 2: Hyperparameters on different models

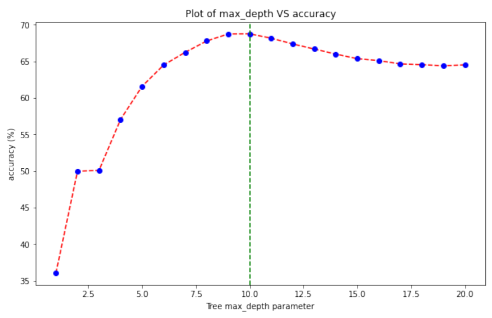

Figure 9: Accuracy vs Max Depth for Decision Tree Classifier

Gaussian Naive Bayes shows the least accuracy out of all. This is because of close proximity of the probabilities of various class labels which leads to a greater number of wrong classifications.

In the decision tree classifier, we use grid search to determine the best parameter value for max_depth. As can be observed in the plot (Figure-6), setting max_depth = 10 yields the highest accuracy with 'gini' as node-splitting criterion. Slight drop in accuracy was observed with 'entropy' as the criterion in preliminary experiments.

In the Random forest classifier, we have used grid search to determine the best parameters value for max_depth, max_features (sqrt or log), and the criterion (gini, entropy). Random Forests outperform decision trees because it is an ensemble model which averages out the error of multiple decision trees classifiers and also leverages the concept of bootstrapping.

We also experimented with a few boosting algorithms in order to achieve improved model performance. All of them yielded higher accuracy than the base decision tree model, which corroborates that ensemble learning techniques like boosting help to build a stronger classifier by training a sequence of weak classifiers.

MLP performs pretty well in our case, as it is a very good function approximator and is robust to noise. It is easy to train and can stabilize well with normalized inputs. It works well with non linear data as it can create non linear decision boundaries.

In our case, SVM turned out to be the best model as SVM can efficiently handle non-linear data using the Kernel trick. It has an L2 regularization feature due to which it has good generalization capabilities. Also, SVM is robust as a small change to the data does not greatly affect the hyperplane and hence the SVM.

Out of all the models, the KNN model runs the fastest as it does not have a training phase like all the other models. The KNN model is sensitive to noisy data and outliers, so it is not able to perform really well as compared to SVM, Multiclass logistic, MLP etc.

Table-3 depicts that SVM performs the best out of the 4 models with an accuracy of 0.732 and Gaussian Naive Bayes performs the worst.

Model	Accuracy	Precision	Recall	F1-Score
Multiclass Logistic Regression	0.714	0.689	0.714	0.694
Gaussian Naive Bayes	0.628	0.646	0.628	0.625
Random Forest	0.717	0.7	0.717	0.7
Decision Tree	0.654	0.641	0.654	0.65
ADA Boost	0.705	0.70	0.70	0.69
Gradient Boost	0.72	0.71	0.72	0.71
XG Boost	0.69	0.7	0.7	0.69
Multi-layered Perceptron	0.717	0.696	0.717	0.699
Support Vector Machine	0.732	0.724	0.732	0.715
K Nearest Neighbours	0.687	0.674	0.687	0.666

Table 3: Results on Testing Set of all models

Model	5-Fold Accuracy
Multiclass Logistic Regression	0.702
Gaussian Naive Bayes	0.61
Random Forest	0.71
Decision Tree	0.649
ADA Boost	0.7
Gradient Boost	0.715
XG Boost	0.687
Multi-layered Perceptron	0.71
Support Vector Machine	0.729
K Nearest Neighbours	0.68

Table 4: Average Test set accuracy using 5-Fold Cross Validation

6 Conclusion and Future prospects

We find that performing a 10-class classification works better than training two separate models for horizontal and vertical positions as there is no penalty term associated with a wrong combined label. We deployed various Multiclass classifiers which have different advantages and disadvantages. Multiclass Logistic Regression, SVM, Boosting Classifiers, and SVM are able to give a decent performance on the given dataset. We learnt about insights of each model by analysing their performance with each other.

We aim to deploy the above models to help out managers and people in the football domain so as to make better decisions.

Member contribution:

Pritish Gulati - Data Visualization, Random Forests, Various Boosting Algorithms, Lit. Reviews.
Pulkit Piplani - Data Preprocessing, Multi-Class Logistic Regression, KNN, Lit. Reviews.
Shayna Malik - Data Preprocessing, Naive Bayes, MLP, Lit. Reviews.
Varun Khurana - Data Visualization, Decision Trees, SVM, Lit. Reviews.

References

[1] https://www.kaggle.com/stefanoleone992/fifa-21-complete-player-dataset

[2] https://www.ijrte.org/wp-content/uploads/papers/v9i1/A2747059120.pdf

[3] https://homepages.dcc.ufmg.br/fabricio/download/lssa_fifa_CR.pdf

[4] https://en.wikipedia.org/wiki/Association_football_positions

[5] https://arxiv.org/pdf/1802.04987.pdf

[6] https://www.researchgate.net/publication/313450788_Towards_Data-Driven_Football_Player_Assessment

[7] https://arxiv.org/abs/1907.00503