PREDICTING AND ANALYZING FOOTBALL OUTCOMES WITH MATCH AND PLAYER STATISTICS

USING SEVERAL MACHINE LEARNING ALGORITHMS

JASPER REMMEN

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE & SOCIETY
AT THE SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
OF TILBURG UNIVERSITY

STUDENT NUMBER

2002643

COMMITTEE

Michal Klincewicz
prof. dr. Starmans

LOCATION

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science &
Artificial Intelligence
Tilburg, The Netherlands

DATE

June 25, 2022
Word count: 8771

PREDICTING AND ANALYZING FOOTBALL OUTCOMES WITH MATCH AND PLAYER STATISTICS

USING SEVERAL MACHINE LEARNING ALGORITHMS

JASPER REMMEN

25/06/2022

Abstract

This thesis will look into the predictions of football matches based on two kinds of statistics, match and player statistics. Not only a high accuracy is important in this research, but understanding the underlying features is essential for translating the results into real football tactics. Based on this, the main research question is: *To what extent can player and match statistics be used as variables in the machine learning algorithms; decision tree, random forest, and gradient boosting to predict the outcome of a football match and explain the result?* With a literature review, the random forest and gradient boosting algorithm were chosen to achieve the best prediction outcomes, and the decision tree was chosen for the analyses of the features. Several databases were created with statistics related to different positions in the field. Better accuracies as the literature provided were found with the gradient boosting algorithm. The three most important variables that were identified by calculating the information gain are touches inside the defensive area, assists and aerials lost.

CONTENTS

1 Introduction 3
1.1 Societal relevance 3
1.2 Academic relevance 3
2 Research Questions 3
2.1 Research gap 4
3 Literature review 5
4 Method 11
4.1 Data 11
4.2 Preprocessing 11
4.3 Models used 13
4.4 Hyperparameter tuning 14
4.5 Evaluation 16
4.6 Error analysis 17
4.7 Packages used 18
5 Results 19
5.1 Model comparison 19
5.2 Feature analyses 22
5.3 Error analysis 28
6 Discussion and conclusion 32
6.1 Discussion 32
6.2 Limitations and future directions 33
6.3 Conclusion 34
A Appendix A 38
B Appendix B 39
C Appendix C 40

1 INTRODUCTION

1.1 *Societal relevance*

The prediction of sports outcomes has always been of interest to many entities. The sports world is a multi-billion industry so the stakes are very high. Football clubs like FC Barcelona, Real Madrid, Liverpool, and Manchester United each have a yearly budget of over 500 million euros. The biggest and most lucrative tournament, the EUFA Champions League, has over two billion euros in prize money. For that reason, clubs try to innovate in different ways to win a piece of this prize money. The use of data science is one of the more recent trends in football. Clubs try to select their players by analyzing data and tracking player movements on the field. For competitive reasons, they don't share these results with the public. It is important that the results can be explained because then the results can be applied in practice. If black-box algorithms, like neural networks, are used the variables might not be explainable and therefore the clubs can't make use of the data.

1.2 *Academic relevance*

Many researchers try to find the holy grail in predicting football outcomes for several reasons. First of all, a reason can be personal interest in the sport. Besides that, football is a game, and sometimes strange, unexpected results happen. An explanation can be that one team has an exceptionally good day while the others have a bad day. Other explanations include luck and tactics. By looking at the data, the piece of the game which is not determined by talent or luck can be identified. Even aspect's that are considered luck may be explained by the data, in a way that ordinary football fans might not see. This also works on other levels. For example, certain features may be identified that make the difference between a good or a great team. For academic research, it is very interesting to look at the available data to identify the parts of football that actually can be explained with data.

2 RESEARCH QUESTIONS

The main research question of this paper is:

> *To what extent can player and match statistics be used as variables in the machine learning algorithms; decision tree, random forest, and gradient boosting to predict the outcome of a football match and explain the result?*

To answer this research question, two sub-research questions will be answered.

sub RQ1 *Which algorithm, decision tree, random forest, or gradient boosting, can predict match outcomes the best using both match and player statistics?*

sub RQ2 *What player statistics should be used as predictors of the match outcome and how can it be explained that they influence the outcome?*

2.1 *Research gap*

In the literature, there is a gap in the prediction of football matches by the features that are used. The most common features that are used are match statistics (Haruna, Maitama, Mohammed, & Raj, 2021), (Alfredo & Isa, 2019) or player statistics (Stübinger, Mangold, & Knoll, 2019), (Cintia, Giannotti, Pappalardo, Pedreschi, & Malvaldi, 2015). No research combines these kinds of statistics to generate a more complete set of features. One paper tries to include ratings from the video game FIFA (Baboota & Kaur, 2019), but it would be better to use actual player statistics to get a more reliable result. Therefore, this research will try to fill this gap by combining 2 data sets, one of player statistics and one of match statistics. Eryarsoy and Delen (2019) comes closed by adding some variables about the age of players and their nationality. Furthermore, this research focuses on analyzing the features that are used and tries to explain why they have an impact on the prediction based on tactics.

3 LITERATURE REVIEW

In the field of predicting football outcomes, a lot of research exists. There are researches, like the work of (Haruna et al., 2021), that use simple data and variables that can predict well. They use several machine learning algorithms to predict match outcomes of the English premier league. Data from one season is used and most of the features are based on the performance of the team (features like ranking, average goals for and against, and away or home advantage). They discovered that the K-NN algorithm finds a very high accuracy of 0.84 in comparison to 0.63 of the Naive Bayes and 0.49 of the random forest. This can be explained by the inclusion of the variable ranking, a high ranking means that a team has won more matches than a lower team and therefore is more likely to win. A downside of this research is the limited amount of data that is used.

Player statistics and betting odds were used to build machine learning models to bet on football matches by (Stübinger et al., 2019). In this research, no match statistics were used as features, only player statistics. They used random forest, Support Vector Machines, boosting, and linear regressions. With the help of random forest and boosting, they were able to generate a profit because the accuracies they achieved were 0.81 and 0.79 respectively. All the models had a higher return than betting without using machine learning models. In this research, additional requirements were used to decide whether to bet or not, these are not necessary for this thesis but it shows that player statistics are valuable in predicting match outcomes, especially when the algorithms random forest and gradient boosting are used (Stübinger et al., 2019).

Other player statistics, like passing behavior, can also be used to make predictions about the outcome of a match. Depending on the league in which the predictions are made, different algorithms perform the best. In three out of four leagues investigated, the random forest algorithm gave the highest accuracy (0.55 in Italy, 0.53 in Spain 0.58 in England) while in the other competition, the German Bundesliga, the K-NN algorithm gave the highest accuracy of 0.6. These results were considerably better than their baseline model which gave a 0.45 accuracy (Cintia et al., 2015). What algorithm is the best, can depend on the league. Since this thesis doesn't make a distinction between the different leagues but combines them it makes more sense to use the random forest model because it performs best in 3 out of the 5 leagues that will be used (the French league 1 was not investigated).

Gradient boosting followed by random forests were the best-performing algorithms to predict match outcomes (win, lose or draw) in the Premier League (Baboota & Kaur, 2019). Both algorithms outperformed the Support vector mechanism (accuracy of 0.59 and 0.57 versus accuracy of 0.54) just as in the previous papers. For that reason, this algorithm (SVM) won't be used. A drawback of this research is that it uses match statistics combined with ratings gathered from the video game FIFA. It would be better to use actual player statistics than a rating from a video game.

Something thing that was also noticed, was the predictions regarding draws were a lot less accurate than the win or lose predictions, independent of the algorithm used (Baboota & Kaur, 2019). This phenomenon is also noticed by other researchers like Raju, Mia, Sayed, and Uddin (2020), whose precision of the matches ended in a draw is significantly lower (between 42% and 46%) compared to the precision of a win for the away team (66-72%) and win for the home team (76-85%). Two possible explanations for this underperformance can be given. The first one is that statistically speaking draws occur less and therefore classification algorithms underestimate the number of draws. And secondly, when the classification is made with a quantitative variable, a rounding criterion is used to determine the class. This criterion can be biased towards another class if that class occurs more often (Golia & Carpita, 2019). Because draws are generally predicted less accurately in the literature, the error analyses will be focused on the correct predictions in each class, win, lose or draw, to see what algorithm is better in predicting the draws.

A comparison between tree-based prediction models to predict football outcomes, using only 14 match-based statistics was conducted in the Premier League over a length of 10 seasons. The Random Forest had the best results with an accuracy of 0.69, followed by gradient boosting with an accuracy of 0.68 (Alfredo & Isa, 2019). This work significantly outperforms the work of (Baboota & Kaur, 2019) while the league is the same. The difference is that Baboota and Kaur (2019) uses statistics that are based on a rating and these can be computed in a way that is not representative while Alfredo and Isa (2019) uses actual match statistics. Also, the performance of Cintia et al. (2015) in the Premier League was less with the same algorithm. Therefore, the kind of data that is used is crucial for the success of the predictions. In this thesis, the combination of player statistics, like the passes from Cintia et al. (2015), will be added to the match statistics to see if the predictions can outperform both pieces of research on the same algorithm.

Also in the work of Eryarsoy and Delen (2019) random forest and gradient boosting were the best performing algorithms, with accuracies of respectively 0.746 and 0.745 significantly outperforming the decision tree (0.597) and neural network (0.559). In their data set, there were a significant number of missing values and they give the reason that the tree models can handle the missing values better (even when they are imputed). The databases that will be used, also have some missing data, but these entries will be removed.

Not every research reaches the best results with tree-based algorithms. For example, Tax and Joustra (2015) predicts Dutch football outcomes the best with Naive Bayes and the multilayer perceptron, outperforming a decision tree and random forest. Both achieved an accuracy of 0.547, which was just slightly higher than their random forest which had 0.542.

Since more papers have decision trees and gradient boosting as best performing algorithms, these will be considered the best performing algorithms. The random forest algorithm is an algorithm based on decision trees. It computes several decision trees (this is a hyperparameter that has to be determined by the input of the model) which are trained individually and then bagged together to find the best result. For classification tasks, the best result is based on a majority vote for each branch. Generally speaking, the more trees, the better the classification task. However, after 128 trees, the increase in performance is very minimal (Probst & Boulesteix, 2017).

The most important aspect that distinguishes gradient boosting is that it tries to minimize a specified loss function. It does this by taking a random subset of the full training data and fitting the base learner (tree) on this data. Then the base learner is updated. This process is iterated on the whole data set until each data point is used in a random subset (Friedman, 2002).

Based on the above-mentioned literature review, the following sub-research question is made:

> *Which algorithm, decision tree, random forest, and gradient boosting, can predict match outcomes the best using both match and player statistics?*

The decision tree will be used as the baseline in this research because it is used as a basis for both the random forest and gradient boosting. Both are designed to improve the accuracy of the decision tree by taking a majority vote for the best one or by updating the initial model to minimize the error. Furthermore, the decision tree has the advantage that it can be easily interpreted and visualized to identify the steps that are done to come to the final classification. This is useful for the next step, analyzing the features.

A good selection of features is important to achieve a good prediction result. If not, the accuracies can be significantly lower. For example, a very large database, containing results from over 200,000 matches was used to predict match outcomes and achieved accuracies with a maximum of 0.46 (Elmiligi & Saad, 2022). Compared to the results achieved in the previously mentioned work, these results are significantly worse. An explanation can be found in the fact that their features are not good. They used only several features, including club name, league, date, and home advantage. These statistics are not very explainable for the reason why a specific result would be achieved.

A framework for predicting sports results with machine learning algorithms exists (Bunker & Thabtah, 2019). For feature engineering, using different (sub)sets of features is suggested. The subsets can be based on feature selection algorithms or expert opinions and have to be compared to find the optimal selection of features for classifying the match outcomes. Furthermore, a distinction between match features and external features is made. The match features are statistics of the specific match, while external features are known before the match starts. In this thesis, several subsets of features will be used. For every line in the field, a database will be created with features that are important aspects of players in those positions.

There are five moments of play in a football match, being established offense, the transition from attack to defence, established defence, the transition from defense to attack, and set pieces (Hewitt, Greenham, & Norton, 2016). The framework in which the moments of play are described cannot be entirely translated to this research, but some characteristics will be used to decide what features should be in what database. The transition from defence to attack, and the transition from attack to defence can be considered the midfield database in this paper. Features they assign to offence include goal attempts, number of passes and efficiency of passes, and possession level. For the midfield, they name among others location of turnover, defensive transition speed, number and length of passes, and player density. Defensive characteristics are based on the surface area, the distance between players, and player density.

There are also other ways to identify attributes that belong to a play style. For example, Fernandez-Navarro, Fradua, Zubillaga, Ford, and McRobert (2016) made a distinction between an attacking play style and a defensive play style. For the attacking play style, performance indicators are features that involve possession, passing direction and passing location, crosses, and shots. Defensive performance indicators are characterized by regains in the field. These kinds of variables are present in the player database and will therefore be used to create separate databases which will be linked to the match database.

The usage of external features in many pieces of research is also noticed by (Raju, Mia, Sayed, & Riaz Uddin, 2020). They make a valid point by stating that post-match statistics are not known before a match starts, and can therefore not be used before to predict the outcome. For that reason, they only use statistics that are known before a match starts to predict the outcome, like home and away win percentage, win streak, points gathered previous season and the average amount of goals scored for away or home matches. With these features, they were able to achieve an accuracy of 67% with a decision tree.

However, not knowing the all the data before a match starts doesn't always have to be a problem. For example, averages of statistics from prior matches can be used (Bunker & Thabtah, 2019). The match database that is used consists of data that is used in this thesis is not known before a match, like shots on target. This is not uncommon, for example, Alfredo and Isa (2019) uses match data that is not known before a match. Their feature selection is done based on the results of the random forest, and they select the features that are used in the model that gives the highest accuracy.

The player database will be manipulated in a way in which all the data is grouped per team per year. By doing so, a team average per season will be created. The advantage of having an average per season is that a bad performance or an off day will be canceled out by exceptional good performances. Furthermore, the features that will be identified as good indicators can give a good idea of the tactic that is most effective for winning matches. A downside is that changes of tactic within the season, for example, because a trainer is fired and replaced by a new one, are not shown in the data.

In this research, it is crucial to know what features are the give the best explanation of the outcome. To get to know what features are the most important, several metrics can be evaluated. For the decision tree, this evaluation can be done based on the Gini index or Entropy (information gain). Also, random forests can be used to identify important features. One way of doing that is by looking at the contribution of a variable by counting the occurrences (Palczewska, Palczewski, Robinson, & Neagu, 2013). This can be applied in a variety of ways, for example, Eryarsoy and Delen (2019) uses the frequency of the first split in the random forest model of each variable to determine which features are the most important.

Following the above literature, the following sub-research question arises.

> *What player statistics should be used as predictors of the match outcome and how can it be explained that they influence the outcome?*

This sub-research question can be answered on two different levels. The first level is which database has the best features and the second is what individual features are important. When these features are identified, they can be analyzed to create an explanation.

When both sub-research questions are combined, the main research question, *To what extent can player and match statistics be used as variables in the machine learning algorithms; decision tree, random forest, and gradient boosting to predict the outcome of a football match and explain the result?* can be answered.

This research starts by looking at the big picture and comparing several machine learning algorithms to classify match outcomes. Afterwards, the focus shifts to a smaller level to identify features that can explain the results.

4 METHOD

4.1 *Data*

Two databases will be used in this research. Both databases were found on Kaggle and are freely available to the public. The first database has scores of matches from the top 5 competitions in Europe; the Premier League (England), La Liga (Spain), Bundesliga (Germany), Serie A (Italy), and Ligue 1 (France). The data ranges from 2014 to 2020. Besides the score of the matches, there is also information about match statistics like shots on target, possession, and cards for each match involved. In total, the database has 41 variables including an index and a total number of 12062 entries (matches).

The second database that will be used consists of player statistics. The data ranges from 2014 to 2021. A lot of different statistics about players are in the data set including physical characteristics, defensive, passing, possession, and shooting statistics. In total, there are 232 variables with 21364 entries. Each entry represents a player for one club in one season. Players that switch clubs during a season have two entries for that specific year.

The databases are updated regularly, the last date the databases were downloaded was 11-04-2022.

4.2 *Preprocessing*

First, the match database was pre-processed. There were no missing values in the data set. A sanity check was done to check if the values were logical. By checking the unique values for each variable, it was revealed that there were several dates (eg. 01FEB) entered in the column of Score. These values were removed. However, this specific variable is not used in the analyses. A new variable was computed based on the goals scored by the home team and goals scored by the away team and used instead. If these numbers were equal, the variable W/L/D (win, loss, or draw) would be a 1, meaning the match ended in a draw. If the home team scored more goals than the away team (so the home team wins), the variable is assigned a 2 and if the away scored the most goals (win away team), the value equals 3. This variable that is computed will be the y variable in the analyses.

The second database has more missing values. These values are represented by NaN, meaning that the values of that specific variable are not available to the player. This can easily be explained. Some statistics are only important for a specific position in the field. For example, the variable shots on target might be missing for goalkeepers and defensive players as they are not very likely to come in scoring positions. Therefore, it was decided to interpret these values as a 0. In the database, there was data of players that played in a different league before they joined one of the five big leagues. This data was removed because it is redundant.

The names of some teams were different in both databases. For the next step to merge the databases, the names have to match. Therefore the names of the player database were replaced by the corresponding name in the match database and both names were converted to lower cases. All of this had to be done manually since there was no consistency in the usage of names. To do this, first, all of the unique club names were printed for both databases. Then the names in both databases were compared, if they were not written the same, the name of the team in the player database was replaced with the name in the match database. For example, sevilla was replaced by sevilla fc, köln by 1. fc köln, and manchester utd by man utd. Another variable that was not consistent in both databases was the variable 'year' and 'season'. This variable is in the match database for one specific year, while in the player database this is for a season. To be able to match the databases, the season variable is split into a year1 and year2 variable. All the player data was then grouped per team per year.

As suggested in the literature review by Bunker and Thabtah (2019), four different databases are created from the player data, based on statistics that are important features for each position in the field (goalkeeper, defence, midfield, and attack). The first selection criteria for this is in the name. Some variables in the database have gk (goalkeeper), def (defence), mid (midfield), or att (attack) in the name, which indicates the position in the field where this statistic is measured and this is matched to the correct database. For example, pressures def 3rd, touches mid 3rd, and touches att pen area. Next to this, some variables can be classified as defensive or offensive based on the characteristic of the action. For example interceptions, tackles, and blocked passes can be classified as defensive because they are used to stop the opponent from scoring, while shots on target, average shot distance and carries into the penalty area are offensive attributes since they tell more about the attempts to score goals. These attributes are then assigned to the defense or attack database. Some variables are useful for the defense, midfield, and attack database, because they are important for all these positions in the field, like statistics about passing (distance or feet used) and heading. The midfield database consists of data that is present in both the defensive and offensive databases because, for those positions, players can be either offensive or defensive. The goalkeeper database has no overlap and includes only data with gk in the name with the addition of variables like goal kicks, clean sheets, and pens saved. For a visual representation see figure 1.

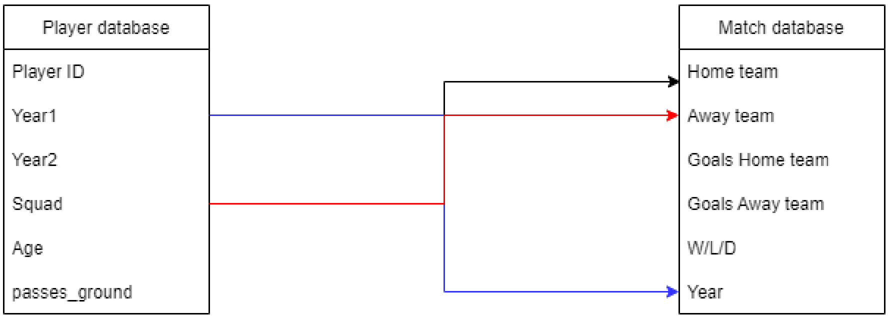

Figure 1: Vizualization of the database merger

After these four databases are created from the player database, these are each merged to the match database on the variables home team and year1. At this point, the database consists of match data and the player data of the home team in each row. To get the full database, the data of the away team is added, again by merging on the name of the away team and year, but this time, to prevent duplicate variables in the same row, the names of the variables that are used for the away team get the prefix away. The final databases, after cleaning and merging, have 7086 entries of which 1784 draws (class 1), 3130 wins home team (class 2), and 2172 wins away team (class 3). From the 12062 matches in the database, 4976 matches are not included. The reason that these are dropped is caused by the number of variables that have a null value. Not all players have all the data present. It was decided that matches that have 20% or more player data values of 0 are removed from the databases.

4.3 *Models used*

Before the models will be applied to the dataset, a train, validation, and test split will be made. The data will be trained on the training set and checked on the validation set. Then the model will be tested on the test set to see if the model over or underfits. The split that is used is a 60/20/20 split, meaning that the training set consists of 60 % of the data, and the test and validation set both have 20 % of the data. After this, hyperparameters can be tuned to increase performance. A random state will be applied to make sure that the split of the train test and validation set is the same every time the model is run. The random state is set at 2. Furthermore, y is stratified to make sure that the proportion of classes in the y variable is the same in all the datasets.

The prediction that is done is a multi-class classification problem of 3 classes. A match can end in a win for the home team, a draw, or a win for the away team. The best performing models in the literature review, will be used in this research as well and be compared with each other. These machine learning algorithms include; decision tree, random forest, and gradient boosting. All of them are used for supervised learning because the outcome of the matches is already known (Tangirala, 2020). The decision tree algorithm is used as the baseline model. This model is very useful because it is interpretable. This characteristic is very useful for the identification of the most important features. By looking at the information gain, the most important features can be identified by looking at the best splits. This is for application in real football situations important. Trainers can focus their tactics on the features that give the highest probability of winning. The random forest is chosen because it trains a large number of decision trees and finds the best prediction by using majority votes. This is useful because a lot of variables are included in the databases and the random forest can generate many possible combinations of variables used in the trees. By looking at what features are frequent in the random forest, important features can be identified. Therefore, this model is useful for this research. Since the classes in the final databases are not distributed equally, the gradient boosting algorithm is used because it performs better on imbalanced databases. By updating the model according to a learning rate to minimize the loss function, the algorithm is less likely to 'favor' the larger classes and is better at predicting the small classes (Tanha, Abdi, Samadi, Razzaghi, & Asadpour, 2020).

4.4 *Hyperparameter tuning*

For the hyperparameter tuning, a grid search will be performed. A dictionary will be provided with the most important hyperparameters and the different values they can take. Then the grid search will iterate over these parameters and create all possible combinations to find the combination which will result in the best outcome for the model (Alibrahim & Ludwig, 2021). The parameters that are found with this grid search, will be used in the final model. More parameters can be tuned as the ones that are mentioned in this section. The ones that are selected, are the most important for either the outcome or for analyzing the features. First, the grid search was applied to only the match database, to check if the amount of variables provided has a feasible computation time. This is necessary because using too many parameters and values causes a high number of variations which can lead to an extensive computation time (Alibrahim & Ludwig, 2021).

For the decision tree, the following hyperparameters are tuned: the criterion on which the split is made (either on Gini impurity or entropy), the depth of the decision tree, the minimum samples that are needed in each leaf and for each split, and the way the split is done (random or at best). For the baseline, the range for depth was set from 2 until 25 with steps of 1, for the minimum amount of samples for the leaves the range was between 1 and 10 with steps of 1, and for the minimum amount of samples for the split the range was set between 2 and 10 also with steps of 1. For the other databases, the range of the depth was decreased to 16 and the step size was increased from 1 to 2 for all these three variables. The maximum amount of features that will be used is left at the default and equals the number of input features.

For the random forest algorithm, the parameters are the same as for the decision tree. However, there is an additional parameter that determines the number of decision trees that should be used in the 'forest' (number of estimators). For the tuning, the number of trees on which the data is trained is set at 32, 64, and 128. Also, it must be decided what number of features will be used. The formula that is used for this can be either the log_2 or the square root of the total amount of features provided. This differs per database and is decided by the grid search to maximize the predictive accuracy. The ranges of the other variables are the same as in the decision tree.

The gradient boosting algorithm has the most hyperparameters to be tuned. Besides depth and the number of samples for the leaf and split, there are also parameters of what loss function should be used and on what criteria the quality of the split must be measured. The loss function can either be deviance or exponential, with deviance as the default setting. Exponential cannot be used in this research, because that requires the predicting value to be of two classes and since there are three classes to predict, the deviance option should be used. The criterion to measure the quality of the split can be done based on the Friedman squared error, or on the mean squared error. Furthermore, the learning rate must be determined, that is the rate at which the model is updated (Friedman, 2002). The learning rate is a default 0.1, for the tuning three additional options will be given being 0.5, 1, and 2. The options for max depth, minimum samples for the split and leaf, and the maximum amount of features, are based on the results of the best hyperparameters of the corresponding decision tree and random forest. This is to increase computational effectiveness. If these assumptions are not made and the ranges are kept the same for the decision tree and random forest, the computation with the grid search would take too long and the kernel could crash. These assumptions can be made because for this model the criterion is the most important together with the learning rate because this has the highest impact on the quality of the predictions. Furthermore, it has been verified by the decision tree and random forest that these parameters are the best for the data set that is used. In Appendix A and B, a full overview is given with the final hyperparameters that are used in the result section.

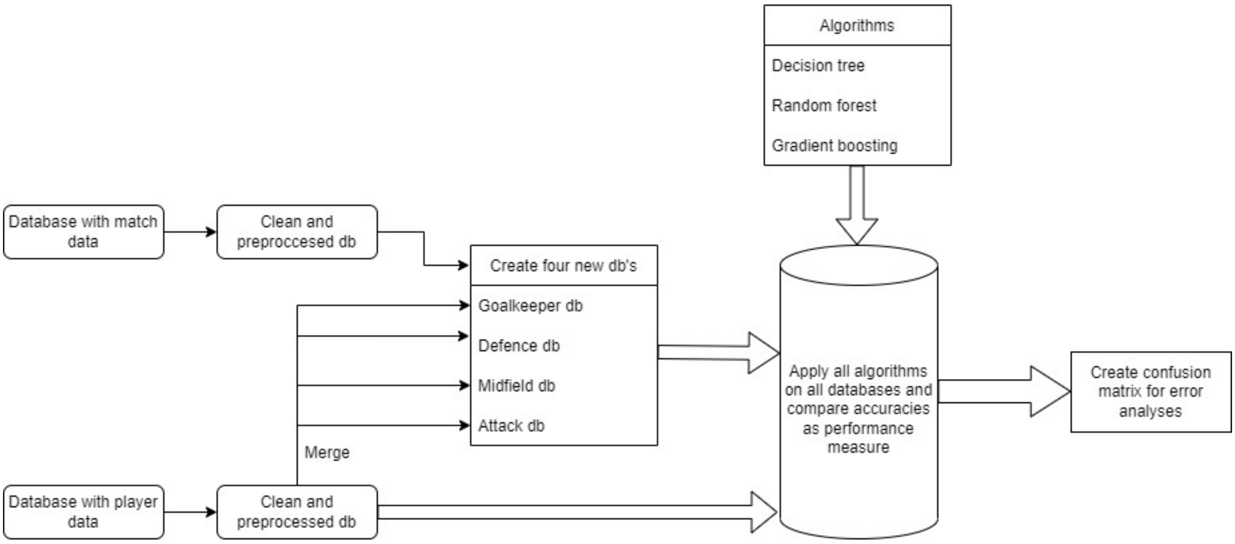
Algorithms
Decision tree
Random forest
Gradient boosting
Database with match data
Clean and preproccesed db
Create four new db's
Goalkeeper db
Defence db
Midfield db
Attack db
Merge
Database with player data
Clean and preproccesed db
Apply all algorithms on all databases and compare accuracies as performance measure
Create confusion matrix for error analyses

Figure 2: Visual representation of the workflow of the model comparison

4.5 *Evaluation*

The metric that will be used to compare the different models is accuracy. Accuracy measures the proportion of correctly classified classes, and can mathematically be expressed as $(TP + TN)/(TP + TN + FP + FN)$ (Tangirala, 2020). This metric is important because it checks how many matches are predicted correctly and can therefore easily identify the best model. Figure 2 shows the workflow for the model comparison.

For the second research question, the information gain will be used to evaluate the features in the decision tree. This can be calculated when the split criteria are based on entropy, which means the variance of the data points at a specific branch. The information gain measures the quality of the split by measuring how much entropy has been reduced by the split of the branch. The entropy of a set of observations can be calculated with the following formula: $-\sum(p_i * log_2(p_i))$, where p_i stands for the probability of the occurrence of a class (Tangirala, 2020). Figure 3 shows the workflow for the evaluation of the features.

For the random forest, the evaluation of the features is done by extracting features that occur the most by applying the function get.support() from sklearn feature selection. This function returns True or False to whether the importance of a feature exceeds the mean importance. By doing so, a robust selection of the most important features can be made.

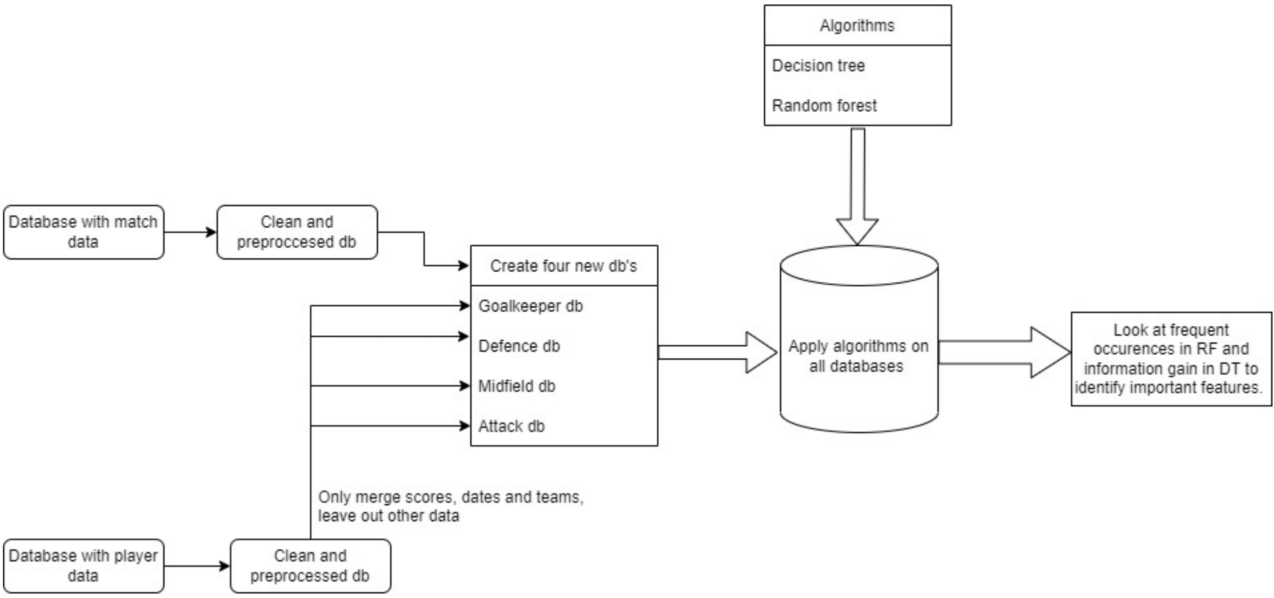
Algorithms
Decision tree
Random forest
Database with match data
Clean and preproccesed db
Create four new db's
Goalkeeper db
Defence db
Midfield db
Attack db
Apply algorithms on all databases
Look at frequent occurences in RF and information gain in DT to identify important features.
Only merge scores, dates and teams, leave out other data
Database with player data
Clean and preprocessed db

Figure 3: Visualization of the process for the feature analyses

4.6 *Error analysis*

The error analyses will be done based on three metrics, precision, recall, and, F1 score. 'Recall is the proportion of real positive classes that are correctly predicted positive' (Powers, 2020). Mathematically this can be written as $TP/(TP + FN)$. The reason that recall is an interesting metric for error analyses is that it represents the proportion that is correctly classified as a fraction of that class. By comparing the recall of different classes, the classes that cause the highest errors can be identified.

'Precision denotes the proportion of predicted positive classes that are correctly positive classes' (Powers, 2020). Mathematically this can be written as $TP/(TP + FP)$. This is interesting for the error analyses because it tells something about the correctness of the predicted values. It can tell which class prediction can be done the best or which class prediction causes the most errors.

Finally, the F1 score is computed, which is computed by taking the harmonic mean of precision and recall. This can be expressed with the following formula $F1 = 2 * (precision * recall)/(precision + recall)$ (Powers, 2020). The F1 score is not used for detecting errors, but for the evaluation of the overall performance.

4.7 *Packages used*

In this research, several packages for python are used. Pandas (McKinney et al., 2011) is used to create and manipulate data frames, while numpy (Van Der Walt, Colbert, & Varoquaux, 2011) is used for data processing and manipulating variables. Scikit-learn (Pedregosa et al., 2011) is used for the machine learning algorithms and evaluation of the features. For the visualization of the decision tree and confusion matrix, the mathplotlib (Hunter, 2007) package is used. Furthermore, a piece of code that was used, was retrieved from the internet and is freely available for the public (TwinPenguins, 2019). The piece of code created a confusion matrix of a three-class classification which was used for the error analyses. For the visualization of the boxplots in the result section, the seaborn library was used (Waskom, 2021).

5 RESULTS

5.1 *Model comparison*

The first results achieved are high in comparison to the results achieved in the literature. The best results are achieved with the gradient boosting model with an accuracy of 0.864 on the test set, with the database that has the match data merged with the midfield data. Remarkable is that the addition of the player data, yields lower accuracies when the random forest model is applied. The differences between the different databases on the same algorithm are relatively small compared to the difference in accuracies between the models. The gradient boosting algorithm is the best one in all the databases, while the random forest outperforms the decision tree three times. The baseline is twice not beaten, in these cases (match combined with midfield and attack) the random forest has a lower accuracy than the decision tree.

Database	Accuracy DT	Accuracy RF	Accuracy GB
Match data	0.782	0.831	0.862
Match and goalkeeper	0.792	0.810	0.869
Match and defence	0.797	0.802	0.865
Match and midfield	0.783	0.781	0.865
Match and attack	0.784	0.795	0.861

Table 1: Accuracy scores on validation set

Database	Accuracy DT	Accuracy RF	Accuracy GB
Match data	0.781	**0.819**	0.862
Match and goalkeeper	0.789	0.804	0.862
Match and defence	0.776	0.804	0.860
Match and midfield	0.790	0.771	**0.864**
Match and attack	**0.791**	0.787	0.860

Table 2: Accuracy scores on test set

The decision trees are visualized to look at the individual features. By doing so it became clear that the away team rating was the most predictive feature for all of the models, followed by the home team rating. As can be seen in the boxplots (figure 4 and figure 5), there is a high dependency between the rating and the outcome of a match. Figure 4 shows that an away team with a rating below 6, in most cases results in a win for the home

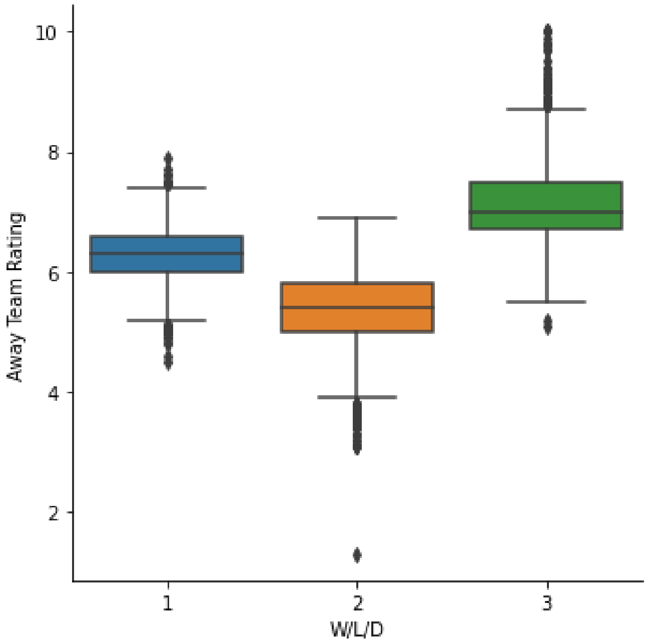

Figure 4: Boxplot of the result of the on the x-axis and the rating of the away team on the y-axis.

team, while the chances of winning an away game increase significantly when the rating is around 7 or higher. For home teams (see figure 5), the chance of winning is the best when their rating is above 6.5 or higher. The reason this is slightly lower than for away teams can be caused by the home advantage. Again, a rating below 6 results mostly in wins for the other team. This feature can be considered a good predictor because it indicates what team is better, but for the feature analyses, this variable is not useful because it doesn't tell anything about specific factors that determine the match outcome. Furthermore, it is not clearly defined in the database how the ratings are computed. Therefore, the whole process, including hyperparameter tuning is repeated to create a model without these two features to find out what player or match statistics are explanatory for the outcome of a match.

After removing these variables, the accuracy dropped significantly, as can be seen in table 3 and table 4. The reason for this drop is the fact that rating is a very good indicator of the better team that is more likely to win. The baseline is only once not beaten, by the match and attack database with the random forest. Furthermore, the random forest outperforms the gradient boosting algorithm once. That is the case in the database where no player data is added.

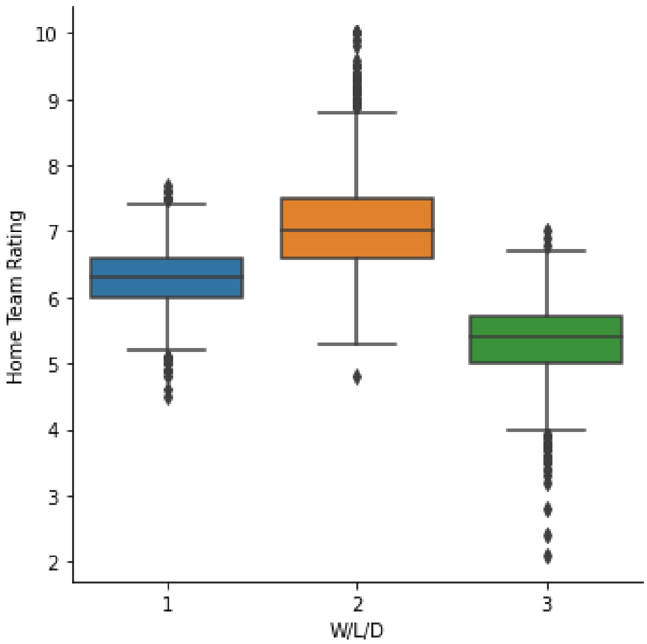

Figure 5: Boxplot of the result of the on the x-axis and the rating of the home team on the y-axis.

Database	Accuracy DT	Accuracy RF	Accuracy GB
Match data	0.586	0.622	0.627
Match and goalkeeper	0.591	0.642	0.646
Match and defence	0.552	0.623	0.651
Match and midfield	0.584	0.612	0.661
Match and attack	0.590	0.622	0.659

Table 3: Accuracy scores on validation set after removing the rating variables

Database	Accuracy DT	Accuracy RF	Accuracy GB
Match data	0.588	**0.632**	0.620
Match and goalkeeper	0.597	0.605	0.635
Match and defence	0.571	0.616	0.627
Match and midfield	0.584	0.611	**0.647**
Match and attack	**0.603**	0.602	0.644

Table 4: Accuracy scores on test set after removing the rating variables

The answer to sub-research question 1 is gradient boosting, in nine out of 10 comparisons. The random forest model performs the best once. If the variables home team rating and away team rating are included in the database with match and midfield features, the predictions are the most accurate. The rating has a significant influence on the result. When these are removed, the best prediction's accuracy drops by 21.7%. The same model stays the best. The addition of player data to the match data is not in all cases an improvement. In some cases, it yields a lower accuracy. For the random forest model, it is in neither case beneficial to add the player data.

5.2 *Feature analyses*

To get a more in-depth view of the important player statistics, the match statistics are removed. The player statistics per club are linked to the results. As can be seen in table 5 and table 6, again the accuracy dropped. The features that are linked to attacking are the best predictors while features about goalkeepers are the worst predictors.

Database	Accuracy DT	Accuracy RF	Accuracy GB
Goalkeeper	0.484	0.495	0.499
Defence	0.483	0.510	0.526
Midfield	0.517	0.524	0.524
Attack	0.531	0.534	0.511

Table 5: Accuracy scores on validation set after removing the match data

Database	Accuracy DT	Accuracy RF	Accuracy GB
Goalkeeper	0.467	0.489	0.493
Defence	0.469	0.496	0.502
Midfield	0.502	0.506	0.500
Attack	**0.509**	**0.511**	**0.509**

Table 6: Accuracy scores on test set after removing the match data

For a more robust first impression of the most important features, the features that are most used in the random forest are extracted, see table 7. Something remarkable is that passing statistics are most frequently used in the defence database, while they are also present in the midfield and attack database, but these have a higher variety and include more features about different aspects. This suggests that defence variables can be less predictive of the match outcome.

Database	Features more used that the mean
Goal-keeper	goals against gk, shot on target against, clean sheets, away goals against gk, away goal kicks, away def actions outside pen area gk, away shots on target against, away clean sheets
Defence	passes completed long, passes completed short, passes completed medium, fouls, crosses, passes ground, passes low, passes right foot, carry distance, away interceptions, away passes completed long, away passes completed short, away passes completed medium, away fouls, away crosses, away passes ground, away passes right foot, away carry distance
Midfield	assists, passes into penalty area, fouls, offsides, passes ground, shots total, shots on target, goals per shot, away passes completed short, away passes completed medium, away assists, away passes into penalty area, away fouls, away crosses, away touches mid 3rd, away touches att pen area, away carries into penalty area, away shots total, away shots on target
Attack	assists, passes into penalty area, fouls, shots on target, goals per shot, away passes completed, away assists, away touches att 3rd, away shots total, away shots on target,

Table 7: Features that occur more than the mean in the random forest algorithm

To get a more detailed insight into the importance of the features, for each decision tree one path is visualized (figure 6 up to and including 9). The path is chosen by looking at each split and taking the one with the highest information gain, which is represented in the triangle. For the interpretation, it is important to keep in mind that all the variables with the prefix 'away' are per season statistics for the away team, while features that don't have this prefix are for the home team. In two cases, the defence and midfield database, the final classification is 100% correct for the specific path. A nuance must be made that the sample size in the final step is relatively low.

In Appendix C, the boxplots for all the variables that are mentioned in the decision tree paths are displayed. The three most important ones, with the most information gain, are displayed in figure 10. Most thresholds lie in the second or third quartile and are therefore not very interesting to discuss deeper. Some thresholds are in the fourth quartile, these are the ones of away clean sheets, away shots on target, passes right foot, away shots total, and assists. This means that the cut-off can be limited when

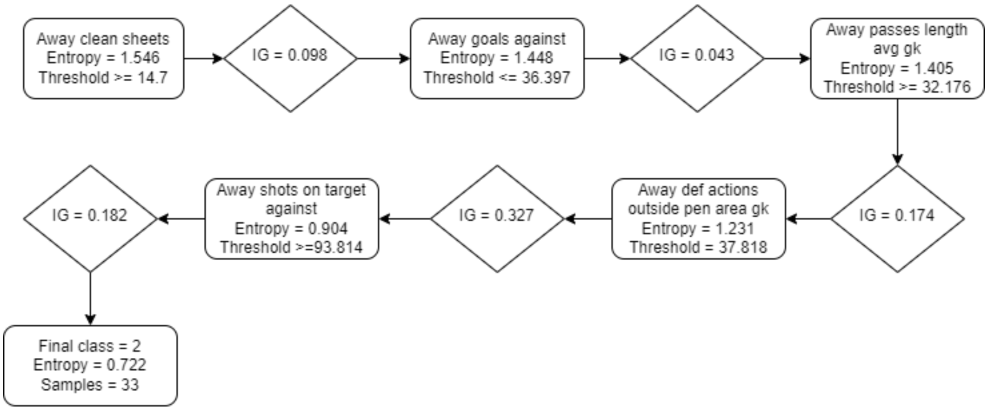

Figure 6: Visual representation of path with the highest information gain in the decision tree for the goalkeeper database

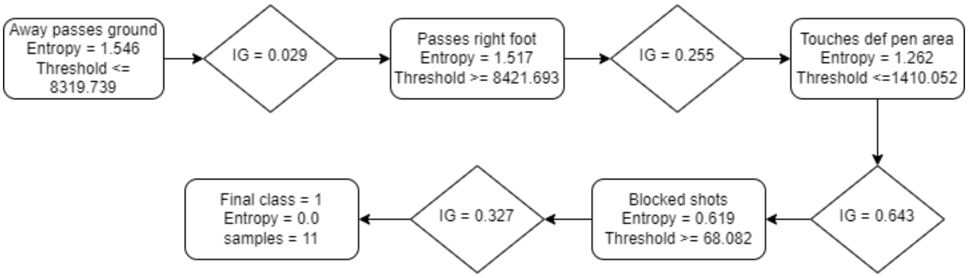

Figure 7: Visual representation of path with the highest information gain in the decision tree for the defence database

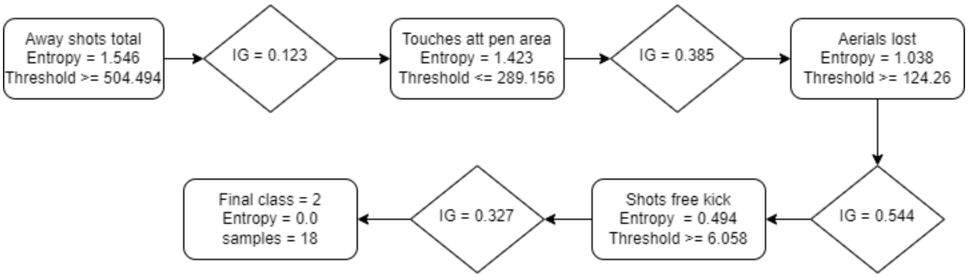

Figure 8: Visual representation of path with the highest information gain in the decision tree for the midfield database

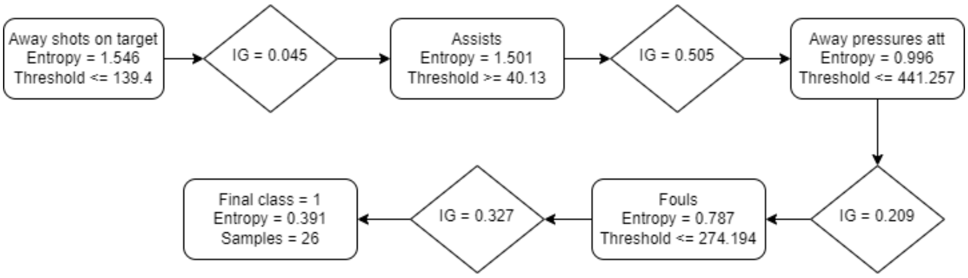

Figure 9: Visual representation of path with the highest information gain in the decision tree for the attack database

cases that are lower than the threshold pass through, depending on the depth of the variable inside the tree and how far the threshold is in the quartile (for example place 76 or 96). For the assist, this is the other way around, because cases higher than the threshold are selected. Therefore a lot of cases are cut off resulting in a high information gain.

To answer the second sub-research question, the three variables with the highest information gain will be further looked upon. Some descriptive information about these variables is presented in table 8.

	Minimum	Maximum	Mean	Standard deviation
Touches def pen area	0	2763	1017.13	994.33
Assists	0	75	21.97	15.57
Aerials lost	0	978	229.13	239.62

Table 8: Descriptive statistics most important features

The split with the highest information gain is the split after touches in the defensive penalty area. Therefore, this is an important feature. Limiting the number of touches in your defensive penalty area can significantly increase the winning chances. When a defender blocks a shot inside the penalty area, it is crucial to play the ball fast outside the penalty area (preferably with the right foot) to limit the number of touches in the penalty area. By keeping the ball too long inside the penalty area, the chances increase that the opponent reconquers the ball in a dangerous position.

As mentioned before, assists have a high information gain. Not every goal has an assist, players can score from individual actions. A lot of assists can therefore either be achieved by scoring a lot of goals or by good team play. Furthermore, the combination of high pressure on the attacking half of the away team in combination with a lot of fouls for the home team combined has a high information gain and makes the home team more likely to win. This can be explained when the away team pressures the

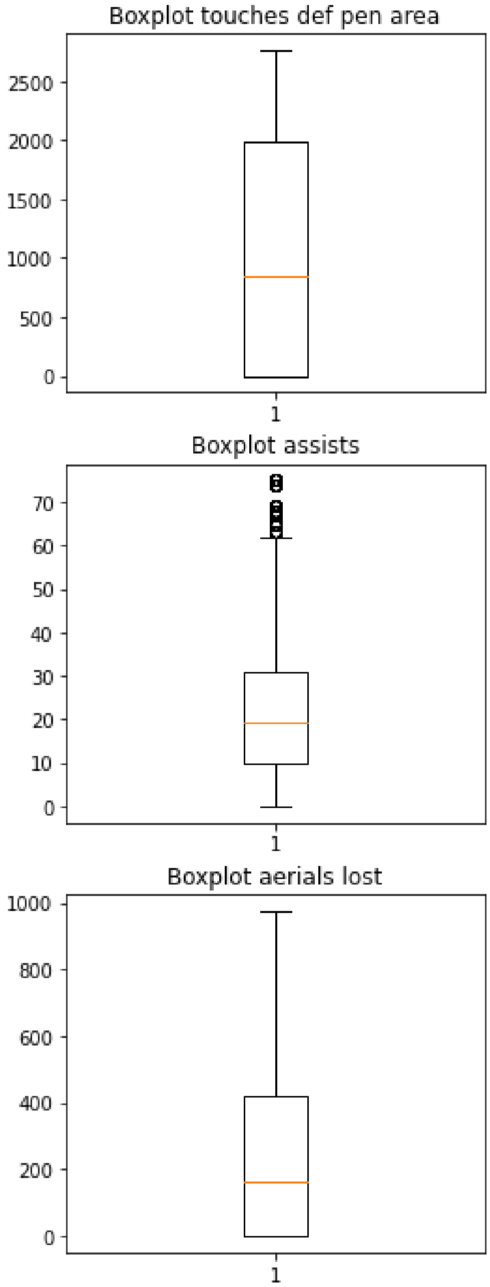

Figure 10: Boxplots of the three variables that create the highest information gain

home team on their half and they try to stop them by playing hard and committing fouls, this pressure can be contained.

The last feature with a relatively high information gain are the aerials lost. This threshold is relatively low, in the second quartile, and exceeding this number, significantly increases the chances of losing a match, especially when the touches inside the opponents' penalty area are below average (that threshold is around the median). Failing on both winning headers and having the ball in your opponents' penalty area appears to be deadly for home teams. The link with shots from free kicks is not very logical and cannot be directly linked with the other variables in this path to identify a play style.

5.3 *Error analysis*

For the error analysis, a confusion matrix was applied to the databases in which the rating variable was removed (tables 9, 10, and 11) and also on the databases that were used for the feature analyses (tables 12, 13, and 14). These matrices are used to see how the classes of the test set were predicted in comparison with their true class. For both models, class two (win for the home team) is the class that is predicted the best, while class one is predicted the worst. For class one the precision is higher than the recall meaning that the chance that when a draw is predicted, this is truly a draw is higher in comparison to the chance that a true draw is classified correctly. While for class two, the recall is higher than the precision in all cases. An explanation can be that the wrongly classified draws are classified as a class two, which drops the precision of this class. Because the absolute values of recall are the highest for class two, it means that these classes are the easiest to detect.

In the model used for the feature analyses, all random forests and the decision tree of the defence, midfield, and attack database, are not able to predict any draws. This is one of the causes that the overall accuracy is not that good. The other predictions of class one are also significantly lower than the predictions of class two and three, meaning that draws are way harder to predict. By looking at the precision of class two predictions in both tables 10 and 13, the gradient boosting algorithm always outperforms the random forest and decision tree, while the random forest outperforms the gradient boosting algorithm on the recall score. The random forest classifies fewer draws correctly (low recall on class one) and classifies them as wins for the home team, which is why that precision is lower while recall is high.

With the F1-score, the models can be compared on the harmonically averaged precision and recall. What is remarkable is that in table 13, five databases score the highest F1-score. The gradient boosting algorithm has the highest F1-score in 8 out of 12, the random forest three times, and the decision tree once. Therefore, also in the error analyses, it appears that the gradient boosting algorithm is the best algorithm for football outcomes with match and player data, especially for the predictions of draws.

	Precision	Recall	F1 score	Support
Match DT	0.4	0.09	0.14	603
Match RF	0.43	0.13	0.2	603
Match GB	0.38	0.23	0.29	603
GK DT	0.39	0.22	0.28	357
GK RF	0.46	0.13	0.2	357
GK GB	0.39	0.23	0.29	357
Def DT	0.44	0.14	0.21	357
Def RF	0.43	0.16	0.23	357
Def GB	0.38	0.27	0.32	357
Mid DT	0.4	**0.35**	**0.37**	357
Mid RF	0.45	0.14	0.21	357
Mid GB	0.39	0.23	0.29	357
Att DT	0.42	0.23	0.3	357
Att RF	**0.47**	0.15	0.23	357
Att GB	0.41	0.23	0.3	357

Table 9: Error analyses class 1, databases with match data but without the rating variables

	Precision	Recall	F1 score	Support
Match DT	0.65	0.8	0.72	1075
Match RF	0.66	0.87	0.75	1075
Match GB	0.69	0.81	0.74	1075
GK DT	0.65	0.81	0.72	626
GK RF	0.64	**0.9**	0.75	626
GK GB	0.71	0.85	0.77	626
Def DT	0.62	0.85	0.71	626
Def RF	0.66	0.87	0.75	626
Def GB	**0.73**	0.82	0.77	626
Mid DT	0.7	0.72	0.71	626
Mid RF	0.66	0.87	0.75	626
Mid GB	**0.73**	0.86	**0.79**	626
Att DT	0.65	0.81	0.72	626
Att RF	0.64	0.88	0.74	626
Att GB	**0.73**	0.87	**0.79**	626

Table 10: Error analyses class 2, databases with match data but without the rating variables

	Precision	Recall	F1 score	Support
Match DT	0.55	**0.72**	0.62	735
Match RF	0.62	0.7	**0.66**	735
Match GB	0.63	0.68	**0.66**	735
GK DT	0.6	0.6	0.6	435
GK RF	**0.64**	0.66	0.65	435
GK GB	0.63	0.65	0.64	435
Def DT	0.56	0.57	0.57	435
Def RF	0.6	0.63	0.62	435
Def GB	0.62	0.64	0.63	435
Mid DT	0.57	0.6	0.59	435
Mid RF	0.6	0.66	0.63	435
Mid GB	0.63	0.68	0.65	435
Att DT	0.6	0.61	0.6	435
Att RF	0.6	0.62	0.61	435
Att GB	0.61	0.66	0.63	435

Table 11: Error analyses class 3, databases with match data but without the rating variables

	Precision	Recall	F1 score	Support
GK DT	0.36	0.03	0.6	357
GK RF	0.00	0.00	0.00	357
GK GB	0.27	0.05	0.08	357
Def DT	0.00	0.00	0.00	357
Def RF	0.00	0.00	0.00	357
Def GB	0.28	0.04	0.07	357
Mid DT	0.00	0.00	0.00	357
Mid RF	0.00	0.00	0.00	357
Mid GB	0.34	**0.10**	**0.15**	357
Att DT	0.00	0.00	0.00	357
Att RF	0.00	0.00	0.00	357
Att GB	**0.38**	0.06	0.10	357

Table 12: Error analyses class 1, databases without match data

	Precision	Recall	F1 score	Support
GK DT	0.49	0.88	0.63	626
GK RF	0.49	0.90	0.63	626
GK GB	0.51	0.81	0.63	626
Def DT	0.47	**0.92**	0.62	626
Def RF	0.50	0.90	0.64	626
Def GB	**0.55**	0.83	**0.66**	626
Mid DT	0.52	0.86	0.65	626
Mid RF	0.53	0.87	**0.66**	626
Mid GB	**0.55**	0.82	**0.66**	626
Att DT	0.53	0.79	0.63	626
Att RF	0.53	0.88	**0.66**	626
Att GB	**0.55**	0.83	**0.66**	626

Table 13: Error analyses class 2, databases without match data

	Precision	Recall	F1 score	Support
GK DT	0.52	0.30	0.38	435
GK RF	0.53	0.30	0.38	435
GK GB	0.48	0.38	0.36	435
Def DT	0.51	0.21	0.30	435
Def RF	0.54	0.35	0.42	435
Def GB	0.51	0.44	0.47	435
Mid DT	0.51	0.43	0.47	435
Mid RF	**0.55**	0.47	0.51	435
Mid GB	0.53	0.46	0.49	435
Att DT	0.47	0.50	0.48	435
Att RF	**0.55**	0.45	0.50	435
Att GB	**0.55**	**0.51**	**0.53**	435

Table 14: Error analyses class 3, databases without match data

6 discussion and conclusion

6.1 *Discussion*

The main research question of this paper is; *To what extent can player and match statistics be used as variables in the machine learning algorithms; decision tree, random forest, and gradient boosting to predict the outcome of a football match and explain the result?* The findings, based on the results presented in the previous chapter are that using the gradient boosting algorithm in combination with the match statistics and midfielder player database with a rating variable included, predicts the outcomes of matches the best. The player statistics, on the other hand, are better indicators of what tactic is responsible for the result with the help of visualized decision trees.

To answer the first sub-question *Which algorithm, decision tree, random forest, or gradient boosting, can predict match outcomes the best using both match and player statistics?*, a pre-selection was done in the literature review. Based on prior work of (Stübinger et al., 2019), (Cintia et al., 2015), (Baboota & Kaur, 2019), (Alfredo & Isa, 2019), (Eryarsoy & Delen, 2019) the random forest and gradient boosting algorithms should perform the best. Because of the high amount of features, the random forest is useful because many random trees are trained with a high variety of combinations and the gradient boosting algorithm handles the imbalance in classes better (Tanha et al., 2020). These two algorithms were compared with a decision tree, which was selected based on the convenience of interpreting the results to answer the second research question. In the predictions made for this paper, the best results came from the gradient boosting algorithm with match and midfield data with an accuracy of 0.861 after hyperparameter tuning, on the test. This significantly beats the baseline prediction, where an accuracy of 0.781 was achieved with only the match data on the test set. This score was also significantly higher than the accuracies found in the literature, mostly caused by the rating variable. After removing this variable, the best results were achieved on the same database also with the gradient boosting algorithm. However, the accuracy dropped to 0.647. The predictions of a match ending in a draw gave the most errors. These were caused by a low recall, meaning that most draws are not noticed by the algorithms and classified as a win for the home or away team. The difficulties with classifying draws correctly was also noticed by (Baboota & Kaur, 2019) and (Raju, Mia, Sayed, & Uddin, 2020).

For the second research question: *What player statistics should be used as predictors of the match outcome and how can it be explained that they influence the outcome?*, the framework provided by (Bunker & Thabtah, 2019) was applied and several databases were created for four positions in the field.

The three features that provided the largest information gain are touches in the defensive penalty area, with an information gain of 0.643, aerials lost with an information gain of 0.544, and assists with an information gain of 0.505. Home teams should avoid too many ball contacts inside their penalty area, especially when they block a ball, they should pass it fast outside their penalty area. Assists are a good indicator of winning a match, with team play this can be achieved. Furthermore, when the away team puts a lot of pressure with their attack, it is beneficial to play hard and make more fouls. Finally, a combination of losing more than average aerials and not having the ball above average in the opponent's penalty area is a good indicator of losing.

6.2 *Limitations and future directions*

A limitation of this research is that in the player database, transfers that occur in between the regular season are not accounted for, so some players might not be correctly assigned to the correct matches. The error of this depends on how much time they played for each club, for that reason it can be that the total amount of time played is larger for some clubs than for others. This error can be solved by manually splitting the values of players between the clubs that the player plays for in that season. First of all, for this to work an additional source needs to be found that contains all this player data and then each variable needs to be split and assigned to the right club and player. Although in-season transfers are less common than pre-season transfers, the amount of work for this can be very labor-intensive considering the high amount of variables, and clubs involved.

Another limitation is the bad predictability of the draws, especially by the random forests classifier. Future research can focus on finding better ways to predict draws. This can be done by using different algorithms or by analyzing what variables can be used to improve these predictions. Another option is to make this problem into a binary classification problem, in which draws are assigned to a either a win or loss. This allocation can for example be done based on the ranking of a club. Draws count as a win for lower-ranked teams because they need the points, while for high-ranked teams draws may feel like a loss and these will count as a loss.

Another interesting future research could be to look even more in-depth into the features. A way to do this is by splitting the analyses per league to find out if different leagues have different aspects that are important for that competition. Another advantage of doing this is that the databases consist of fewer null values. Most null values are caused because a specific feature is not available for a certain competition or season. Therefore, by splitting the analyses per country, certain features can be removed that are not available for one league, while they still can be used for another one.

By adding results from international matches from the Champions League, Europe League, and Conference League, different tactics can be compared to each other to find out which is the better one.

For future research, the created databases could be enriched. This can be done in multiple ways. First of all, data from more leagues can be added, for example by collecting the same data for clubs in the Dutch Eredivisie or the Belgium Jupiler Pro League. Besides adding more leagues, data for a longer period can be collected, either by finding data that goes back further in history or by adding data of newly played matches.

6.3 *Conclusion*

The goal of this thesis was to find a better way to predict football than was already done and to provide useful insights into tactics that can lead a team to victory. The accuracies found in the literature could be beaten by incorporating a variable rating which tells what team is usually the better one. Without this variable, the predictions become a lot less accurate falling in between the results that already exist (some better, some worse). The feature analyses provide three features that have a high information gain and are important for winning or losing a game, being touches inside the defensive area, assists, and aerials lost. Knowing these facts, managers can adjust their tactics and train on certain skills like heading.

REFERENCES

Alfredo, Y. F., & Isa, S. M. (2019). Football match prediction with tree based model classification. *International Journal of Intelligent Systems and Applications*, *11*(7), 20–28.

Alibrahim, H., & Ludwig, S. A. (2021). Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization. In *2021 ieee congress on evolutionary computation (cec)* (pp. 1551–1559).

Baboota, R., & Kaur, H. (2019). Predictive analysis and modelling football results using machine learning approach for english premier league. *International Journal of Forecasting*, *35*(2), 741–755.

Bunker, R. P., & Thabtah, F. (2019). A machine learning framework for sport result prediction. *Applied computing and informatics*, *15*(1), 27–33.

Cintia, P., Giannotti, F., Pappalardo, L., Pedreschi, D., & Malvaldi, M. (2015). The harsh rule of the goals: Data-driven performance indicators for football teams. In *2015 ieee international conference on data science and advanced analytics (dsaa)* (pp. 1–10).

Elmiligi, H., & Saad, S. (2022). Predicting the outcome of soccer matches using machine learning and statistical analysis. In *2022 ieee 12th annual computing and communication workshop and conference (ccwc)* (pp. 1–8).

Eryarsoy, E., & Delen, D. (2019). Predicting the outcome of a football game: A comparative analysis of single and ensemble analytics methods.

Fernandez-Navarro, J., Fradua, L., Zubillaga, A., Ford, P. R., & McRobert, A. P. (2016). Attacking and defensive styles of play in soccer: analysis of spanish and english elite teams. *Journal of sports sciences*, *34*(24), 2195–2204.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, *38*(4), 367–378.

Golia, S., & Carpita, M. (2019). On the improvement of soccer match result predictions. *Data Science & Social Research 2019 Book of Abstracts*, 72.

Haruna, U., Maitama, J. Z., Mohammed, M., & Raj, R. G. (2021). Predicting the outcomes of football matches using machine learning approach. In *International conference on informatics and intelligent applications* (pp. 92–104).

Hewitt, A., Greenham, G., & Norton, K. (2016). Game style in soccer: what is it and can we quantify it? *International Journal of Performance Analysis in Sport*, *16*(1), 355–372.

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in science & engineering*, *9*(03), 90–95.

McKinney, W., et al. (2011). pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, *14*(9), 1–9.

Palczewska, A., Palczewski, J., Robinson, R. M., & Neagu, D. (2013). Interpreting random forest models using a feature contribution method. In *2013 ieee 14th international conference on information reuse & integration (iri)* (pp. 112–119).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, *12*, 2825–2830.

Powers, D. M. (2020). Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*.

Probst, P., & Boulesteix, A.-L. (2017). To tune or not to tune the number of trees in random forest. *J. Mach. Learn. Res.*, *18*(1), 6673–6690.

Raju, M. A., Mia, M. S., Sayed, M. A., & Riaz Uddin, M. (2020). Predicting the outcome of english premier league matches using machine learning. In *2020 2nd international conference on sustainable technologies for industry 4.0 (sti)* (p. 1-6). doi: 10.1109/STI50764.2020.9350327

Raju, M. A., Mia, M. S., Sayed, M. A., & Uddin, M. R. (2020). Predicting the outcome of english premier league matches using machine learning. In *2020 2nd international conference on sustainable technologies for industry 4.0 (sti)* (pp. 1–6).

Stübinger, J., Mangold, B., & Knoll, J. (2019). Machine learning in football betting: Prediction of match results based on player characteristics. *Applied Sciences*, *10*(1), 46.

Tangirala, S. (2020). Evaluating the impact of gini index and information gain on classification using decision tree classifier algorithm. *International Journal of Advanced Computer Science and Applications*, *11*(2), 612–619.

Tanha, J., Abdi, Y., Samadi, N., Razzaghi, N., & Asadpour, M. (2020). Boosting methods for multi-class imbalanced data classification: an experimental review. *Journal of Big Data*, *7*(1), 1–47.

Tax, N., & Joustra, Y. (2015). Predicting the dutch football competition using public data: A machine learning approach. *Transactions on knowledge and data engineering*, *10*(10), 1–13.

TwinPenguins. (2019). *Three class confusion matrix.* `https://datascience.stackexchange.com/questions/40067/confusion-matrix-three-classes-python`. (Accessed May 7, 2022)

Van Der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The numpy array: a structure for efficient numerical computation. *Computing in science & engineering*, *13*(2), 22–30.

Waskom, M. L. (2021). Seaborn: statistical data visualization. *Journal of*

Open Source Software, *6*(60), 3021.

A APPENDIX A

Hyperparameters

	DT	RF	GB
Match	criterion: gini, depth: 10, leaf: 7, split: 9, splitter: random	criterion: entropy, depth: 12, features: sqrt, leaf: 1, split: 2, estimators: 128	criterion: friedman mse, learning rate: 0.5, loss: deviance, depth: 10, features: sqrt, leaf: 1, split: 9
Match and goalkeeper	criterion: gini, depth: 10, leaf: 9, split: 6, splitter: best	criterion: entropy, depth: 12, features: sqrt, leaf: 3, split: 2, estimators: 128	criterion: squared error, learning rate: 0.2, loss: deviance, depth: 10, features: sqrt, leaf: 9, split: 6
Match and defence	criterion: gini, depth: 8, leaf: 9, split: 2, splitter: best	criterion: gini, depth: 14, features: sqrt, leaf: 3, split: 8, estimators: 128	criterion: friedman mse, learning rate: 0.2, loss: deviance, depth: 8, features: sqrt, leaf: 9, split: 8
Match and midfield	criterion: entropy, depth: 4, leaf: 1, split: 2, splitter: best	criterion: gini, depth: 12, features: sqrt, leaf: 1, split: 2, estimators: 128	criterion: squared error, learning rate: 0.2, loss: deviance, depth: 4, features: sqrt, leaf: 1, split: 2
Match and attack	criterion: gini, depth: 4, leaf: 1, split: 2, splitter: best	criterion: gini, depth: 10, features: sqrt, leaf: 1, split: 6, estimators: 128	criterion: squared error, learning rate: 0.2, loss: deviance, depth: 4, features: sqrt, leaf: 1, split: 6

Table 15: Hyperparameters found with grid search and applied to for the first model comparison.

B appendix b

Hyperparameters

	DT	RF	GB
Match	criterion: entropy, depth: 8, leaf: 7, split: 5, splitter: random	criterion: entropy, depth: 12, features: log2 , leaf: 1, split: 4, estimators: 128	criterion: squared error, learning rate: 0.1, loss: deviance, depth: 8 , features: log2, leaf: 7, split: 5
Match and goalkeeper	criterion: gini, depth: 6, leaf: 9, split: 6, splitter: best	criterion: entropy, depth: 10, features: sqrt, leaf: 5, split: 8, estimators: 128	criterion: squared error, learning rate: 0.05, loss: deviance, depth: 6, features: sqrt, leaf: 8, split: 9
Match and defence	criterion: entropy, depth: 6, leaf: 5, split: 4, splitter: random	criterion: entropy, depth: 14, features: sqrt, leaf: 3, split: 2, estimators: 64	criterion: squared error, learning rate: 0.1, loss: deviance, depth: 6, features: sqrt, leaf: 5, split: 2
Match and midfield	criterion: gini, depth: 4, leaf: 1, split: 2, splitter: best	criterion: entropy, depth: 12, features: sqrt, leaf: 7, split: 6, estimators: 64	criterion: squared error, learning rate: 0.1, loss: deviance, depth: 4, features: sqrt, leaf: 1, split: 2
Match and attack	criterion: gini, depth: 6, leaf: 9, split: 2, splitter: best	criterion: gini, depth: 10, features: sqrt, leaf: 5, split: 2, estimators: 64	criterion: squared error, learning rate: 0.05, loss: deviance, depth: 6, features: sqrt, leaf: 9, split: 2

Table 16: Hyperparameters that are used in the model comparison after removal of the rating variables

c appendix c

Boxplots

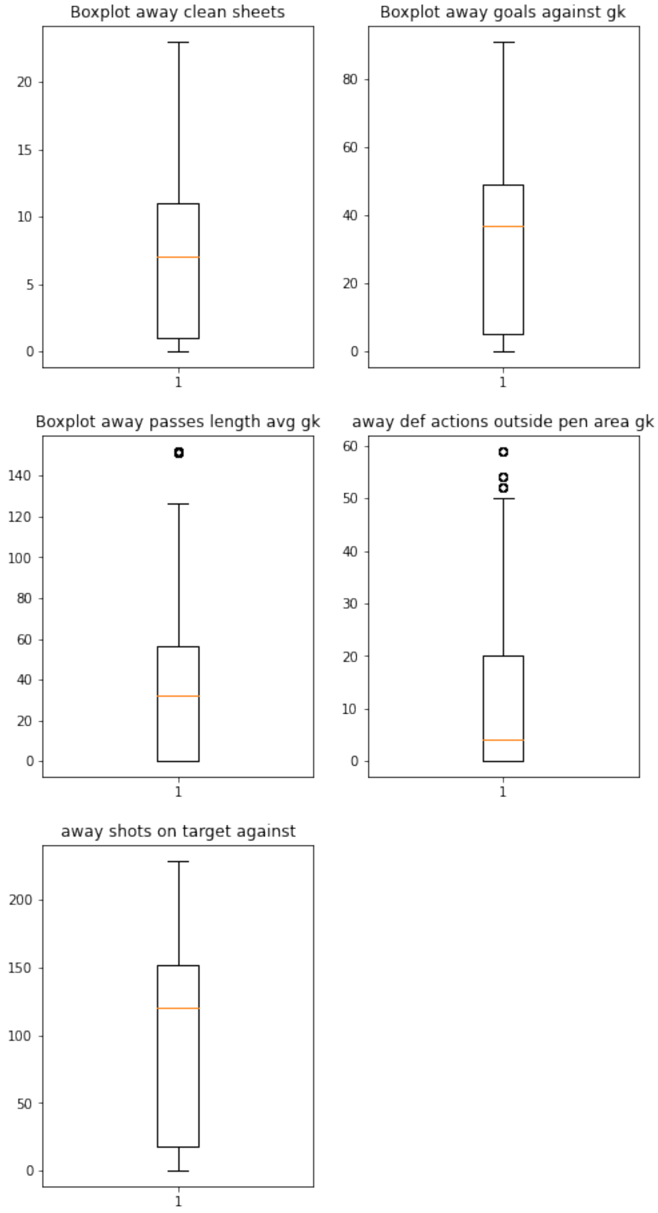

Figure 11: Boxplots of the variables that are used in the feature analyses of the goalkeeper database

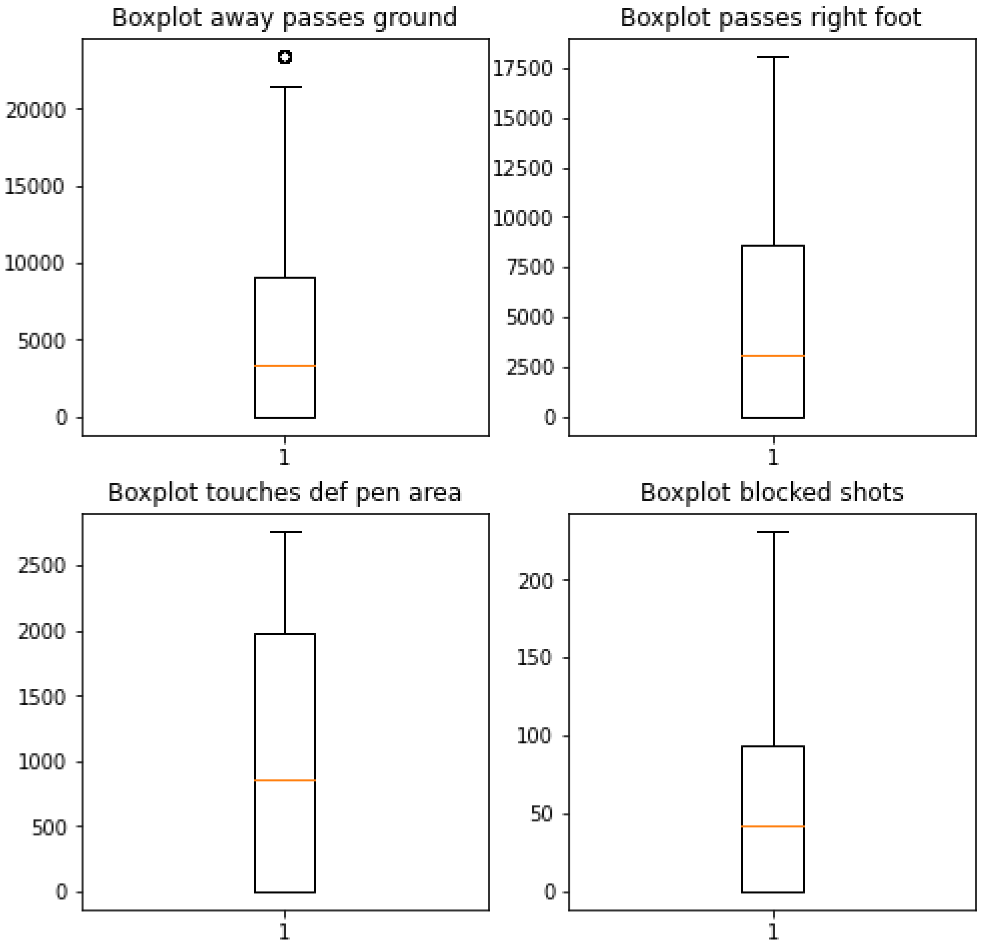

Figure 12: Boxplots of the variables that are used in the feature analyses of the defence database

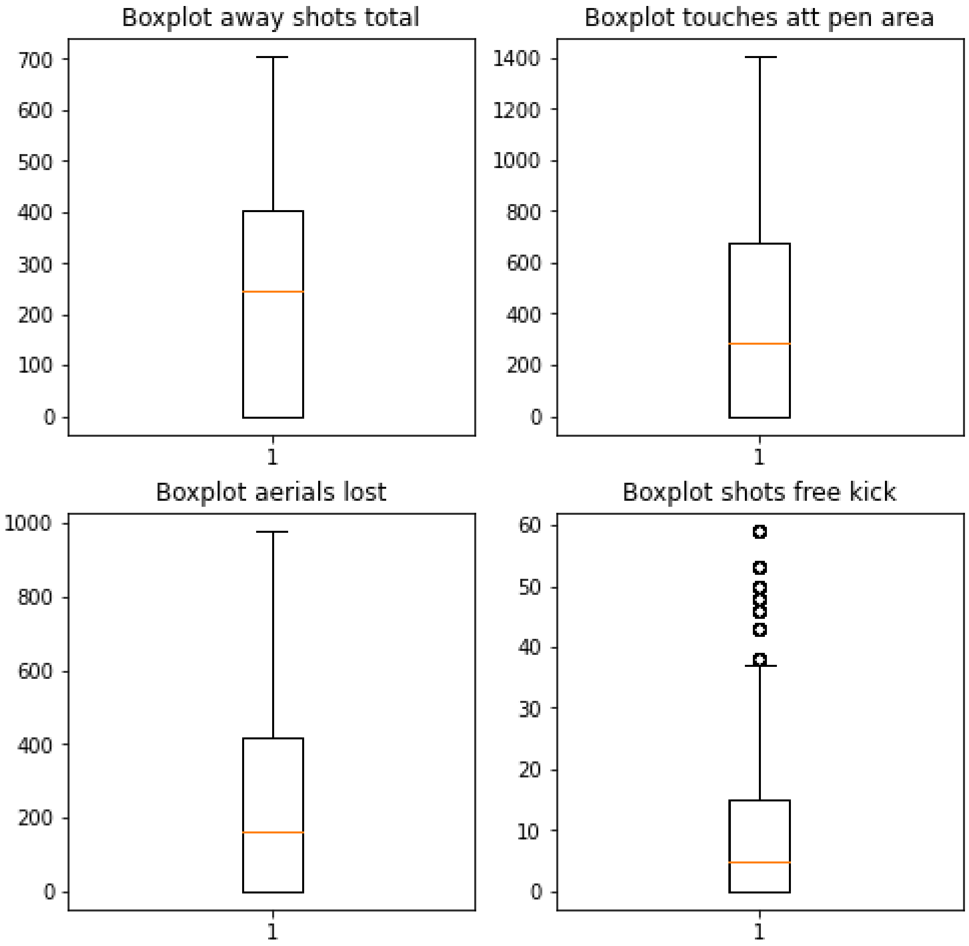

Figure 13: Boxplots of the variables that are used in the feature analyses of the midfield database

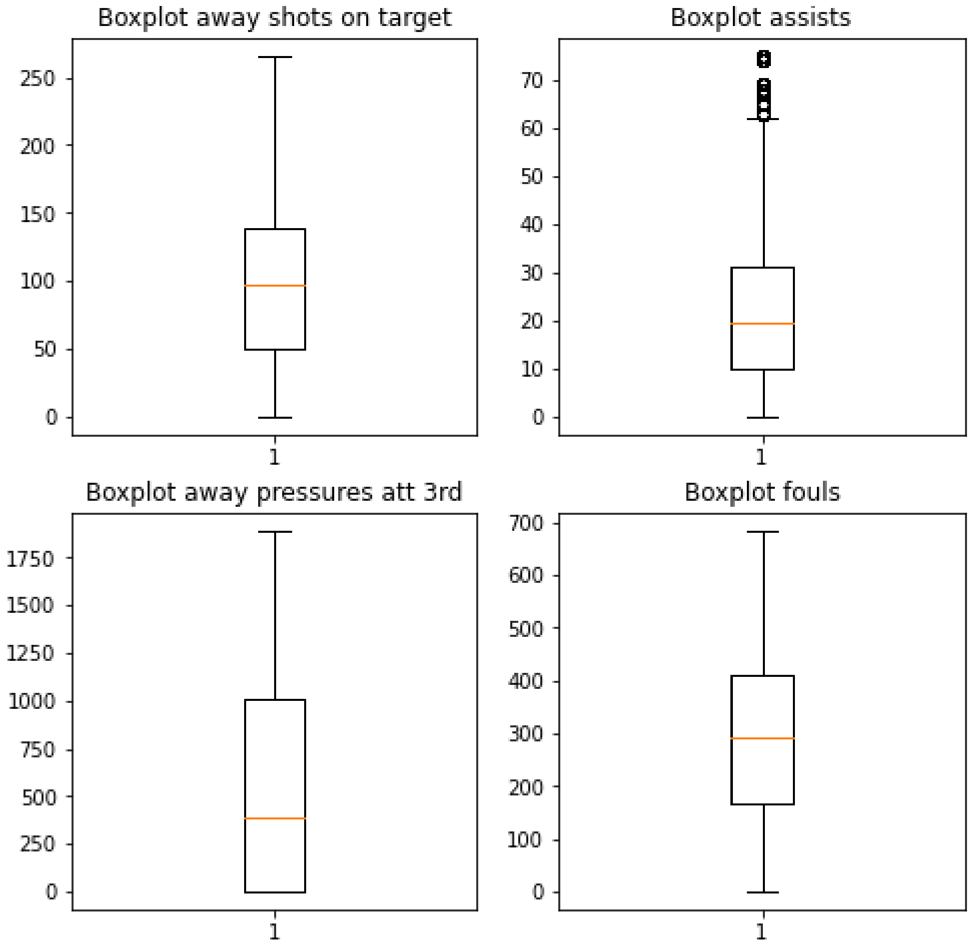

Figure 14: Boxplots of the variables that are used in the feature analyses of the attack database