Data Mining 1:
Predicting positions of soccer players
Project Report

Team No. 9

presented by
Jurgen Amedani (1628022), Cara Maria Damm (1631263), Paul Hesselmann (1371380), Allwyn Menezes (1671634), Martin Mlinac (1364487), Carmen Rannefeld (1631070)

submitted to the
Data and Web Science Group
Prof. Dr. Bizer
University of Mannheim

1 Application Area and Goals

With soccer being the most popular sport worldwide it is not surprising that a good player is very expensive and that the soccer clubs are investing a lot of effort into the development of young talents [1]. One important aspect in the development of young talents is to identify the ideal position, so the player can exploit his full potential. The question that arises is, are there specific attributes a goalkeeper, defender, midfielder and striker need to have? Is an ideal goalkeeper very tall and is it important that a defender has a stocky body type?
One way to evaluate if pattern exist is to analyze current soccer player and the position they are fulfilling. To do this we are using the FIFA 19 Dataset which is available on Kaggle.
FIFA 19 is a popular football simulation game based on the world class soccer player which was developed and released by Electronic Arts in September 2018. The three main challenges for Data Scientists are collecting a significant amount of training data, cleaning and organizing the data as well as applying models [2]. Each challenge is time consuming which results into high cost.
In this paper we will evaluate different classification approaches, apply prepossessing methods an perform a comparison in terms of accuracy. For prepossessing the data, attributes were cleansed, some excluded and aggregated. A detailed description can be read in chapter 3. By applying different classification approaches an identification of the best approach to predict a player's field position for unseen records is expected.

2 Structure and Size of the Data

In this project, which focuses on soccer analytics, the FIFA 19 dataset from Kaeggle is used. It includes information about every player who is registered to the SoFIFA database [3]. The database includes more than 18207 registered players, for each player 86 different attributes are available. The attributes can be grouped in ten groups. Table 1 contains the attributes for each group: In the data set one player is assigned to exactly one position out of 25 available positions. As players might play on different positions during a season or training, the potential for 24 positions (except goalkeeper position) is recognized as well in the data set. The position is written down as abbreviation meaning: These position potentials are not applied for goalkeepers. Not all the attributes have filled values. One example is the *loaned_from* field which does not contain values for all the players since not every player is loaned from another club. Another reason for the missing information is that some players are young and not all the information has been collected yet. Analyzing the data set showed that there are 48 players were only key attributes like personal information are provided.
Attributes of the statistical groups are continuous data with a range from 0 to 100. Other attributes are categorical data. Furthermore, the data set includes numerical attributes like wage, release clause, weight and height. The wage and release clauses start with a value of 1000 while the specialty attribute is not smaller than 731.

Personal Information	Database ID, Name, Age, Nationality, Height, Weight
Contract Information	Club, Wage, Release Clause, Jersey Number, Joined date, Loaned from, Contact valid until
Mental and Physical Capabilities	Aggression, Interceptions, Positioning, Vision, Penalties, Composure
Sports and Soccer Information	Value, International Reputation, Overall Ranking, Potential (correlation to overall ranking), Special, Position (club), Position (national team), Preferred Foot, Week Foot, Skill Moves, Work Rate Defense, Work Rate, Attacking, Body Type, Real Face
Attacking Statistics	Crossing, Finished, Heading Accuracy, Short Passing, Volleys
Skill Statistics	Dribbling, Curve, Free Kick Accuracy, Long Passing, Ball Control
Defending Statistics	Marking, Standing Tackle, Sliding Tackle
Movement Statistics	Acceleration, Sprint Speed, Agility, Reactions, Balance
Goalkeeping Statistics	GK Diving, GK Handling, GK Kicking, GK Positioning, GK Reflexes
Strength Statistics	Shot Power, Jumping, Stamina, Strength, Long Shots

Table 1. Selection of attributes

3 Preprocessing

Not all of the 86 attributes are needed to predict the position of a player therefore we applied the "optimize selection operator" and analyzed the log reports from Rapidminer to select the important attributes. This showed that the following attributes: *sliding tackle, skill move, long passing, heading accuracy, finishing, crossing and sprint speed* were rated with a high relative importance. With no influence the other playing position results were weighted. More details and results of the attribute selection are described in the data mining chapter section 4. Attributes marked as personal and contract information were excluded from the data set as they are not relevant to determine a players position. The data mining processes started with a higher aggregation of player's position which were joined into four groups: *defender, midfielder, striker and goalkeeper*. The following positions were grouped together into the attribute *Position_grouped* replacing the original position attribute:

Striker:
- ST, CF, LF, LS, LW, RF, RS, RW

Midfielder:
- CAM, CDM, LCM, CM, LAM, LDM, LM, RAM, RCM, RD, RM

Defender:
- CB, LB, LCB, LWB, RB, RCB, RWB

Goalkeeper:
- GK

After applying this aggregation 6838 midfielders, 2025 goalkeepers, 5866 defender and 3418 strikers (total 18147) are used to train the models. *Position_grouped* was marked as label in the classification model.
The attributes *release clause* and *wage* are reported in a format including k (thousand) and m (million). These values have been converted. The *height* is converted from foot into centimeter. The *weight* has been recalculated from pound into kilogram. If there was no information provided for *height* and *weight* null values were added which allowed to exclude them by filtering. The attribute *preferred foot* converted into a binomial value, meaning 0 for right and 1 for left. Before the start of the training, examples were a value of *position_grouped* was missing were removed.
During the data mining process, a second data set was prepared to test data and a potential overfit of the model. The test data set based on FIFA data from 2017 included 2652 midfielders, 632 goalkeeper, 2534 defenders and 1131 strikers (total 6949). The FIFA17 data set was also extracted from the SoFIFA.com database two years ago. However, many of attribute names differ from FIFA19 data set which. Therefore Rapidminer could not map the attributes directly and additional preprocessing steps were needed that included upper- and lower case renaming as well as changing abbreviations like "Free-kick accuracy" into "FKAccuracy".

4 Data Mining

To identify the best suited classification algorithm for our problem we applied K-Nearest Neighbors (KNN), Decision Trees, Gradient Boosted Trees, Deep Learning, Supported Vector Machines and Random Forest to the FIFA2019 data set. Our mining process was separated in three phases: First, we ran the "cross validation" operator for each algorithm. Once we had first results for recall, precision and accuracy we transitioned in the second phase, where we aimed to optimize the results for each algorithm. Finally, in the third phase we tested our learned models. To ensure that our models are neither under- nor overfitted we used the FIFA17 data set which varies from the FIFA19 data set. In the following sections the different algorithms and their results from the first and second phase are presented.

4.1 K-Nearest Neighbors

K-Nearest Neighbors (KNN) classifies unseen cases based on the k nearest neighbors. First, the "optimize parameters" operator was applied for values of k between 1 and 100 as well as a 10-fold cross validation. The result of this optimization showed that the ideal value for k = 62 which provides an accuracy of 87.83%. The best performing class was *goalkeeper* with a precision of 100% and the worst performing class was midfielder with a precision of 81.45%. Applying our model to the FIFA17 test data set the results presented in Table 3 could be measured. In phase two of the mining process using KNN we started to optimize

	Training data	Test data
Accuracy	87.83%	87.37%
Weighted recall	88.46%	87.92%
Weighted precision	90.08%	88.86%

Table 2. Results from KNN algorithm on training and test data

the algorithm by optimizing the attributes. To do this we used the "Optimize Selection" and the "Optimize Selection (Evolutionary)" operator which determined the weight for the attributes based on the Gini Index and the Information Gain. The optimal attributes are presented in table 3. By selecting the attributes,

Optimize Selection (9)	Crossing, Finishing, HeadingAccuracy, ShortPassing, Dribbling, LongPassing, Vision, Marking, SlidingTackle
Information Gain (24)	Crossing, Finishing, HeadingAccuracy, ShortPassing, Volleys, Dribbling, FKAccuracy, LongPassing, BallControl, Acceleration, SprintSpeed, Agility, Balance, Jumping, Stamina, Strength, LongShots, Interceptions, Positioning, Composure, Marking, StandingTackle, SlidingTackle, Vision
Gini Index (21)	Crossing, Finishing, HeadingAccuracy, ShortPassing, LongPassing, BallControl, Acceleration, SprintSpeed, Agility, Balance, Jumping, Stamina, Strength, LongShots, Interceptions, Positioning, Composure, Marking, StandingTackle, SlidingTackle, Vision

Table 3. Selection of attributes

the accuracy increased to 88.28 % (Optimize Selection), 88.15 % (Information Gain), 87.97% (Gini Index). Table 4 shows the results on FIFA 2017 data set after the optimize selection.

	Accuracy	Weighted Recall	Weighted Precision
Optimize Selection	87.24%	87.84%	88.56%
Information Gain	87.39%	87.94%	88.95%
Gini Index	87.18%	87.70%	88.65%

Table 4. Results of KNN on test data

As last optimization the "Optimize Parameters" operator was used again, but only for the selected attributes with k values (1-100, 100 steps). This resulted into a value of $k = 57$ with the subset of attributes from the operator, and a $k = 1$ for both Information Gain and Gini Index. With this last optimization the results couldn't be improved anymore and the accuracy dropped by 0.2%.

4.2 Gradient Boosted Trees

After using KNN we continued with algorithms which are using decision trees. First, we uses a classical decision tree but the results could not compete with KNN, therefore we started to apply more advanced algorithm such as the Gradient Boosted Trees (GBT).
The GBT algorithm is used to create an ensemble of decision trees trough gradually improved estimations. The output is a classification model which can be applied to the test data set for a prediction of the label attribute *position_grouped*. Using GBT provides the advantage that a written report about the weights of the attributes with respect to the label attribute is created [4]. The algorithm builds variuos trees which are focusing to include attributes with higher weights than before. The trees are builtt on variations of the original dataset. Classification errors from previous trees are taken into account before building a new tree [5].
To run the algorithm an "optimize selection" operator was applied to test the best combination of maximal tree depth and number of trees. The highest accuracy result was scored building 32 different trees. The maximal depth had no influence according to the operator testing with a step size of 3, since a accuracy of 87 % was achieved regardless of the tree depth of 10 or 25.
The GBT logs the most important variables to build the model. The identified attributes were HeadingAccuracy (percentage 0.21 %), SlidingTackle (0.19 %), Vision (0.17 %) and Finishing(0.1 %). This result for identification of attributes corresponds to one from KNN shown in table 3.
The confusion matrix in table 5 presents the test results. Noteworthy are the bad results for predicting a striker, while all goalkeepers were classified correctly. The overall accuracy was 87.02 %. Weighted recall is 88.15 % and weighted precision is 88.88 %.
Applying the learned model to the test data set an accuracy of 87.06 % was

	True Strikers	True Goalkeeper	True Midfielder	True Defender	Class precision
Pred. Strikers	881	0	245	4	77,96 %
Pred. Goalkeeper	0	629	0	1	99,84 %
Pred. Midfielder	245	2	2183	172	83,9 %
Pred. Defender	5	1	224	2357	91,11 %
Class recall	77.9 %	99.53 %	82.32 %	93.01 %	

Table 5. Confusion Matrix for Gradient Boosted Trees algorithm

scored. As before, the prediction for goalkeepers and defender worked well with a recall higher than 91 %. For the defender class the recall value increased by 1.12 % from 91.89 % to 93.01 %. However a decrease in recall by 0.47 % was measured for the goalkeepers. Using the cross validation the difference for the strikers and the midfielder is less than 2 % compared to the FIFA19 data set.

4.3 Deep Learning

The deep learning (DL) algorithm is based on an multi layer artificial neural network [6] and is one of the most promising machine learning algorithm. DL is making major advances in solving problems that have resisted the best attempts of the artificial intelligence community for many years. It has turned out to be very good at discovering intricate structures in high-dimensional data and is therefore applicable to many domains of science, business and government. Additionally DL is beating records in various areas such as image recognition and speech recognition [15]. Therefore we decided to apply DL to our classification problem. Because DL is a very complex algorithm we used RapidMiners automated modeling functionality for this algorithm. By applying the the learned model to the test data set we were able to score an overall accuracy of 87.78%. The result of the confusion matrix is presented in table 6.

	True Strikers	True Goalkeeper	True Midfielder	True Defender	Class precision
Pred. Strikers	884	0	240	4	78,37%
Pred. Goalkeeper	0	632	0	0	100%
Pred. Midfielder	242	0	2192	138	85,23%
Pred. Defender	5	0	220	2392	91,4%
Class recall	78,16%	100%	82,65%	94,4%	

Table 6. Confusion Matrix for deep learning algorithm

4.4 Support Vector Machines

Support Vector Machines (SVM) is a linear model for classification and regression problems. The SVM Linear takes a set of input data and predicts, for each given input, which of the two possible classes comprises the input, making the SVM a non-probabilistic binary linear classifier [7]. We use cross validation for estimating statistical performance, classification by regression to build a polynomial classification model through regression learner and SVM linear to build a model that assigns new examples into one category or the other.
A change in the value of the parameter C of SVM, which is the complexity constant, sets the tolerance for misclassification in its parameters where higher C values allow for 'softer' boundaries and lower values create 'harder' boundaries [7]. A complexity constant that is large will lead to overfitting and values that are too small may result in over-generalization.
The algorithm looks at the extreme cases and draws a decision boundary known as hyperplane. The result from the optimization provides C for SVM Linear with the best value of 1.0. Training SVM on the FIFA 2019 data set, as shown in the table 7, leads to an the overall accuracy of 85.06%. The best performing class is goalkeeper with 99.95% accuracy and the worst performing class is strikers

with 79.65% accuracy. Precision and recall of the training set are 86.44% and 87.87% respectively. Applying the model to the test data set provides an overall accuracy of 83,74% as shown in table 7. The best performing class is Goalkeeper with 100% accuracy and the worst performing class is Striker with 76.10% accuracy. The precision and recall of the test set obtained are 85.42% and 85.76% respectively.

	Training	Testing
Accuracy	85,06 %	**83,74%**
Accuracy Goalkeeper	99,95%	100%
Accuracy Defender	87,1%	87,9%
Accuracy Strikers	83,57%	76,1%
Accuracy Midfielder	79,65%	79,04%
Weighted Precision	86,44%	85,42%
Weighted Recall	87,87%	85,76%

Table 7. Comparison of results of support vector machine algorithm

4.5 Random Forest

Finally, we applied the random forest classifier as our last classification algorithm. In general, random forests are used in classification and regression problems [9]. A random forest is an ensemble method which consists of a collection of multiple random decision trees. In many data sets, the random forest performs better than the decision tree classifiers [10]. Each tree is generated by random vectors of the training data sets. The nodes in the decision trees are represented by the attributes.
Once the model is applied to new examples, each tree predicts a class by following the branches of the decision tree. The output is a voting classification model, that includes the decision trees in the random forest. Which means, that each tree in this forest makes a decision and the class with the most votes determines the final class. The classification of the random forest varies less than the individual classification of each random tree. This is due to the fact that every classification in the random forest is treated equally [9].
To find the optimal values the "Optimize Parameter" operator ran with different values for number of trees (20 to 90 in steps of 7 with linear scale) and maximal depth (0 to 100 in steps of 10 with linear scale), using the accuracy as splitting criterion. The resulting optimal values for number of trees is 90 and for maximal depth is 50 (as well as: 76 for number of trees and 100 for maximal depth and 90 for number of trees and 50 for maximal depth). For the first combination of values the following results were scored: $accuracy$ = 88.39%, $weightedprecision$ = 90.22% , and $weightedrecall$ = 89.15%. Our best performing class is again the goalkeeper with a precision of 100% and our worst performing class is again the midfielder with a precision of 83.53%. The model

was tested for the three value combinations mentioned above. Besides accuracy also gain ratio, information gain and gini index where used as splitting criterion. The best result was archived with 76 trees and a maximal depth of 100, using information gain as splitting criterion. The results for this value combination are shown in table 8.

Splitting criterion	Accuracy	Weighted Recall	Weighted Precision
Accuracy	84,63%	85,77%	87,13%
Gain Ratio	87,55%	87,67%	89,24%
Information Gain	**88,67%**	**89,25%**	**89,81%**
GINI index	88,62%	89,23%	89,70%

Table 8. Results of random forest model on test data

4.6 Evaluation setup and results

In this section the results of the used algorithms are summarized. To test a possible overfitting the model generated from optimize selection operator was applied to new data from FIFA17 as described earlier.
Table 9 shows the performance scores accuracy, recall and precision for the models. The model built with random forest algorithm performs best on the test data.

	Accuracy	Weighted Recall	Weighted Precision
Gradient Boosted Trees	87,06%	88,19%	88,20%
Random Forest	**88,67%**	**89,25%**	**89,81%**
Support Vector Machines	83,74%	85,76%	85,42%
Deep Learning	87,78%	88,80%	88,75%
KNN	87,37%	87,92%	88,86%

Table 9. Performance scores from test run

Since the classification of a soccer players position is a multi-class problem random forest works better than support vector machines on this problem. Recall and Precision for support vector machines model are the lowest values in comparison with all other models. The deep learning algorithm performed good as well since a lot of training data was available, one main requirement to apply this algorithm.
Starting with the simplest model KNN and decision tree were the first choice. But decision trees are trained on the complete data model, while random forest trees are learned on random samples. This optimize the node boundaries, the diversity of nodes and the robustness of the model [13].

The model of gradient boosted trees performs worse than random forest. According to Ravanshad [14] random forest and gradient boosted trees are different in the way the trees are built. While gradient boosted trees work better on unbalanced data, which was not the given case, random forest are harder to overfit. Less overfitting potential was decisive advantage as the test was executed on an completely unseen data set.

4.7 Discussion of results

The perfect classification of goalkeepers can be justified by strength and weaknesses of these football positions.

Goalkeepers have low values regarding sliding tackle, skill moves, sprint speed, etc. A visualization in a coordination systems 1 shows that goalkeepers (black) have the biggest deficits in these and other attributes (values in the bottom left corner). Defenders (blue) are better in these categories than goalkeepers. While midfielder (green) and strikers (orange) have nearly same strengths (one dot on top of each other in middle and top right corner).

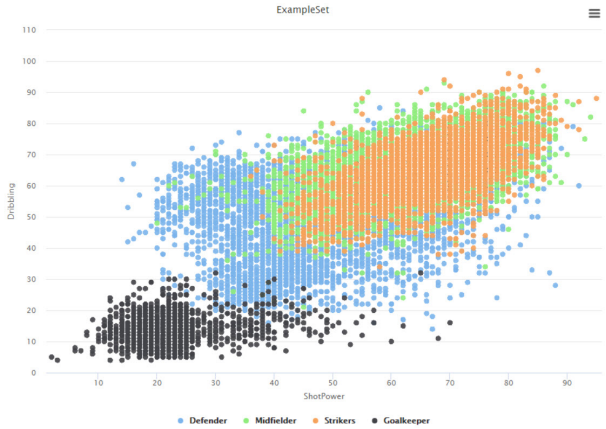

Fig. 1. Visualization of two characteristics: dribbling and shot power

The result matches the expectations as the boundaries between strikers and midfielders are blurry. These two attributes are only an excerpt from more than 80 attributes but are typical for the data set.

As a conclusion, the data model to predict a soccer players position scores good result on the training and test data sets. Due to nearly identical attribute values for midfielders and strikers the accuracy decreased. Certainly possible is that midfielders would be better strikers and the other way around. As further improvement a classification model containing more classes trough a distinction between offensive and defensive midfielders can be considered.

References

1. Die Transfermarkt Top Marktwerte, URL: https://www.transfermarkt.de/1-bundesliga/marktwerte/wettbewerb/L1. Last accessed May, 17 2019
2. Crowdflower 2016 data science report, 2016 URL: https://visit.figure-eight.com/data-science-report.html. Last accessed May, 17 2019
3. Sofifa.com https://sofifa.com/players. Last accessed May, 12 2019
4. Rapidminer Documentation GBT, URL: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/trees/gradient_boosted_trees.html, last accessed May 12, 2019
5. Towards Data Science: Understanding Gradient Boosting Machines, URL: https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab. Last accessed May 22, 2019
6. Rapidminer Documentation Deep Learning, URL: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/neural_nets/deep_learning.html. Last accessed May 23, 2019
7. Rapidminer Documentation SVM, URL: https://docs.rapidminer.com/8.2/studio/operators/modeling/predictive/support_vector_machines/support_vector_machine.html. Last accessed May 22, 2019
8. Rapidminer Documentation SVM Evolutionary, URL: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/support_vector_machines/support_vector_machine_evolutionary.html. Last accessed May 22, 2019
9. Rapidminer Documentation Random Forest, URL: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/trees/parallel_random_forest.html. Last accessed May 22,2019
10. Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. Introduction to Data Mining. Pearson International ed. Boston Munich [u.a., 2006. Print. Pearson International Edition.]
11. Rapid Miner Support Vector Machines, URL: https://docs.rapidminer.com/8.2/studio/operators/modeling/predictive/support_vector_machines/support_vector_machine.html. Last accessed May 20, 2019
12. Rapid Miner Support Vector Machines Evolution, URL: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/support_vector_machines/support_vector_machine_evolutionary.html. Last accessed May 20, 2019
13. Towardsdatascience.com, URL: https://towardsdatascience.com/why-random-forests-outperform-decision-trees-1b0f175a0b5. Last accessed May 23, 2019
14. Ravanshad,Abolfazl. Gradient Boosting vs Random Forest. URL: https://medium.com/@aravanshad/gradient-boosting-versus-random-forest-cfa3fa8f0d80. Last accessed May 23, 2019
15. LeCun Y, Bengio Y, Hinton G. Deep learning. nature. 2015 May;521(7553):436.