Will they score a goal?

VT Department of Electrical and Computer Engineering
ECE 5424: Advanced Machine Learning

Final Project Report

Submitted by:
Radhakrishna Adhikari -rka@vt.edu

Dec 6, 2022

1 Abstract

This project investigated the possible usage of a soccer events dataset to predict whether certain events lead to a goal. The dataset contains over 900,000 events from leagues across Europe. The importance of this problem is highlighted by the increase in sports analytics, not only for betting purposes, but for improving team performance since identifying the best scenarios for scoring a goal can lead to training techniques which aim to replicate those conditions in a match. This classification problem was explored using the machine learning architectures of decision tree classifiers (DTC), random forest classifiers (RFC), multi-layer perceptron (MLP), and support vector machines (SVM). These models were explored and compared with one another by testing the parameters which give us the best model. The further challenge we addressed with this dataset was the target variable imbalance in the amount of events that lead to a goal and the events that do not lead to one. We have explored different balancing techniques such as oversampling and class weighting. In order to optimize our models we also incorporated feature selection. We have also incorporated more features that we hypothesized would lead to greater accuracy in classification in the form of player ratings. All of our experimentation processes are presented in this final report in detail, including the results of using different model architectures, the different experiments we propose which test different balancing techniques and optimization of parameters.

2 Background

Sports analytics is an emerging field and is showing potential in many areas such as sports betting, training methods, and for soccer clubs to make decisions on how to train or what players to buy [1]. Soccer is one of the most popular sports in the world, and one of the reasons for this fascination is the fact that more than 85% of soccer matches end in either a draw or are won by only two or fewer goals [2]. Furthermore, predicting soccer match outcomes is complex due the mix of factors and chance events that play a role. Even in the case of weaker teams playing against stronger teams, the outcome is not easy to predict, since single events such as a red card, or a key player becoming injured, can play a decisive role. Everything considered, we also believe that the outcome of soccer matches are not purely random.

Our project aims to use machine learning techniques to analyze the best possible scenarios for players to score a goal. This is a supervised classification problem since the aim of this project is to assign a label of whether a goal was scored to a number of events that happened preceding the instance of whether it was a goal or not. It is supervised since our dataset already has the outcomes of whether it is a goal and the aim would be to classify future, unknown instances [3]. Goals in soccer are the most important match events because they directly determine the outcome of the match. We can therefore assume that goals scored carry important information such as the winning team being stronger than the losing team based on the goals scored and the margin of victory determining the difference in strength between teams. This classification problem aims to use a dataset of match events to predict whether they would lead to a goal [1].

Previous studies have attempted to predict soccer match outcomes before the matches have occurred. These studies have tried to predict the results based on a betting perspective [4], and some have used machine learning approaches [5]. Another of analyzing soccer data would be to try and develop insights about how teams and individuals on the team are doing by assessing heatmaps aligned with the playing field or computing features of successful actions by players on the field. This project, however, aims to use existing data to do predictions that can be used in the future. The other aspect of this project is to find the best architecture to do these predictions. Our specific aims follow in the next section.

3 Specific Aims

The project has five specific aims:

- **Classify whether the event will lead to a goal:** We will use the key feature variables that we identified as being important to predict whether a goal is scored. It is desirable to know which events lead to goals that could be useful to elite sports teams when making decisions about which players to purchase and how to train to maximize potential to score a goal and to prevent goals being scored.
- **Identify the key features that lead to a goal:** Using the predictive models we aim to analyze, we want to identify the features that contribute the most to the model prediction.

- **Comparing multiple architectures and data balancing techniques:** Compare different model architectures that we are using to predict whether a goal is scored, namely: decision tree classifiers (DTC), random forest classifiers (RFC), multi-layer perceptrons (MLP), and support vector machines (SVM). We shall also be comparing different scenarios of balancing the data such as oversampling and class weight balance.
- **Optimizing the performance of the classifier:** Understand how different architectures performs using the different parameters in our models to get the best model and evaluate the different model performances
- **Evaluate the performance of the classifier:** Incorporating overall player ratings as well as "shooting" rating from the FIFA online database for each player who was involved in a shot in the dataset. This will be used to test how much influence the players ability has on whether the shot will results in a goal or not.

4 Research Design and Method

The problem of predicting the possibility of a goal is formulated as a supervised classification model. There are several classification algorithms that could be helpful in solving this problem. Since, we are trying to build a prediction model that could be useful to coaches for decision making in the game, we tried to pick the model that could be easily explainable to someone without any machine learning background. The decision tree [6]is explainable in very simple terms and the tree could be visualized in a nice manner. In addition, the simple structure of DTC makes it computationally advantageous. Therefore, we picked the decision tree classifier [6] as our first algorithm. In addition, we utilized three other models, RFC, MLP, and SVM, to compare the performance with DTC and with each other.

An RFC model [7] is an ensemble of a number of classified trees, which are fitted on the sub-samples of the dataset. The RFC is the average of these trees, therefore, RFC can control the over-fitting and decrease the variance of the model without increasing the bias. However, the prediction range of an RFC is bound by the range of the train data.

An MLP model [8–10] comprises multiple layers of networks, whose weights are determined through the training process to minimize the loss function. MLP is capable of capturing the complex nonlinear relationship between predictor variables. However, a complex MLP requires many calculations to be done in parallel, which makes the algorithm performance depend on the strength of the hardware. In addition, determining the size of the network is a difficult task.

The SVM [11] algorithm is used to determine the hyperplane that separates the data points with the objection of minimizing the distance between support vectors to that hyperplane. SVM is effective for high dimensional data, however, it does not perform well on data whose classes are not well separated.

In addition to those models, we also did two data balancing techniques. First, We did an oversampling technique where we modified the proportion of the observations with label 0 and 1. We increased the observations with label 1 (goal) and made the equal proportion

of both labels with this technique. Second, we did class weighting technique. Class weighting forces the training method to weight samples with certain class values more (or less) than others.

We implemented the above models in following steps:

1. **Data extraction**

   From the dataset we extracted from kaggle, we first chose the features that might be good predictors of goal. We did this manually. Some of the features in the dataset were not relevant to the prediction of the goal, so we removed those features. For instance, id_odsp and id_event gave us the unique identifier of the game and event respectively. These columns were not relevant to our analysis, so we dropped them from our dataset. Many other features were dropped which we elaborate in the next section.

2. **Data pre-processing**

   From preliminary exploratory analysis, we observed that there are several observations with missing data. In most of the entries, more than 50% of the predictor features had missing values. Thus, removing the missing values was the best option for our model. We removed observations with missing data. In addition, we converted the "player" column which has the name of a player making the shot into the column with the respective FIFA player ratings. We then split the dataset into training and test dataset.

3. **Dataset balancing**

   From the preliminary analysis, we found that we have an imbalanced dataset, and we could see its effect in the model. The model was predicting almost all the test set observations to be no goal which had a higher number of observations. Therefore, we implemented two dataset balancing techniques before running the model.

4. **Build different models**

   We built models on the final dataset for the given dataset balancing method. We implemented 4 different algorithms as described above; DTC, RFC, MLP, and SVM.

5. **Feature selection**

   Out of the shortlisted predictor variables, we ran a sequential feature selection algorithm to choose the best variables for the model. We ran the sequential feature selector algorithm in sklearn for each model to find the best set of variables for each model. The performance of models saturated after adding 3 best features for all models. Therefore, we selected three best features for each model.

6. **Evaluate the model performance**

   We evaluated the model performance at the end. We checked confusion matrices, overall accuracy scores , F_1 score, Precision score, and Recall score.

5 Dataset

The project uses a single dataset obtained from Football Events [12] to train and test the model. The dataset is derived by reverse engineering the text commentary of events that take place in the game using regex. The dataset provides information of 9,074 games, totaling 941,009 events from 5 European leagues [12]. Although the Dataset contained numerous features, our team has decided to use a certain subset of features to apply the model in aiding managers of football teams to assist them in making key decisions while taking a shot which would possibly lead to a goal.

Most of the features that were removed were primarily due to two factors. Firstly, some of them added no meaning to the model such as IDs of the events which are not very useful in predicting the target variable. Secondly, some of the features could have brought bias into the dataset which included features that clearly dictated that a goal had been scored or dictates that there is no possibility to score a goal. To deal with this problem, our team removed those features which brought any kind of bias to the model. Two features had some categories which brought bias to the dataset, but some categories were quite useful to predict the model. For those two features - shot_outcome and shot_place, we removed the certain categories that brought bias to the dataset and kept the remaining categories in the dataset. The target feature of the model is 'is_goal' which is a binary variable. In addition to using this dataset, our team compiled another feature: Player Rating which consists of the rating of players involved in an event. Our team made a separate code to match the names of the players with the rating of the player found on a website which contained ratings of all professional soccer players [13]. The predictor features used in the model are listed in Table 3 below.

6 Experiments

The model was developed in small phases and each experiment helped in the overall development of the project. In total three experiments were conducted in the development of this project that are described in detail below.

- **Experiment 1** In Experiment 1, after the data had been pre-processed, the new dataset was left with 32403 rows which was an acceptable amount of data. The dataset was quite imbalanced before preprocessing with about a 20/80 split of the target variable. After pre-processing of the dataset, the target variable had a lower imbalance ratio of 5/7 which was later solved with balancing techniques. This gave the team a perspective on how challenging it is to deal with different datasets and optimize it to yield the best results for the model. In lieu of dealing with an imbalanced dataset, two methods are adopted - oversampling and class weights to balance the dataset. We compared different balancing techniques and used the best strategy which yielded the best performance for each model. In addition to balancing the dataset, we explored feature selection on the dataset to be able to understand the features being used in the dataset. Since, most of the features used in the dataset are categorical features. The team applied one hot encoding to the selected feature to make a logically correct model.

Predictor Features	Type of Variable	Purpose of variable
Location	Categorical Variable	Approximate location of event on the Football Field.
Time	Continuous variable	Gives the minute of the game
Shot_Placement	Categorical Variable	Placement with respect to the goal post.
Shot_outcome	Categorical Variable	The outcome after the shot is taken.
Bodypart	Categorical Variable	The body part used by the key player in the event
Assist_method	Categorical Variable	Type of assist received by the key player.
Situation	Categorical Variable	The situation in which the event has taken place.
Fast_break	Binary Variable	Turnover of possession of the ball.
Player Rating	Continuous Variable	Rating of the key player in the event ranging from 0 to 100
Shot Rating	Continuous Variable	Rating of the power of the shot taken by the player

Table 1: Feature description table

- **Experiment 2** After the dataset was pre-processed, balanced and one-hot encoded, the goal of the second experiment was to compare the performance of the balanced and unbalanced dataset on all of the proposed architectures i.e. DTC, RFC, MLP, and SVM classifiers. The key differentiating factor of comparison for this experiment would lie in the dataset itself. This would give the team a chance to understand the importance of balancing datasets, feature selection and one-hot encoding. The performance of all the proposed architecture would be evaluated using metrics such as accuracy score, F_1 score, recall score, precision score and confusion matrix. The accuracy score is defined as the number of correct predictions over the number of trials. The precision, recall and F_1 scores are calculated using the formulae listed below:

$$Precision = \frac{t_p}{t_p + f_p}$$

$$Recall = \frac{t_p}{t_p + f_n}$$

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$$

where, t_p is true positive, f_p is false positive, and f_n is false negative.

The visualization of the prediction outcome of the classification model is presented by the confusion matrix. Once baseline models were built, the team then explored different techniques to optimize the proposed architectures by tuning the parameters of each architecture. The MLP manually optimized such as finding the optimum number of hidden

layers for the MLP whereas some architectures such as SVM and DTC were optimized using the GridSearchCV algorithm. The team optimized the model for both the balanced and unbalanced dataset and recorded the performance of the model on the metrics discussed above.

- **Experiment 3** The goal of the third experiment was to add two new predictor features: Overall rating quantifies the overall skill level of the player on a scale of 1 to 100 and Shot rating quantifies shooting skill of the player on a scale of 1 to 100. The primary reason for adding these ratings was to help the team manager make more informed decisions in their training techniques by analyzing ratings of the players involved in the key event. The team thought that the player making the shot might be a very important predictor to predict the goal. For instance, there is a substantial difference between Lionel Messi (world renowned soccer player) making the shot vs Sagar Thapa (an average soccer player that is not as famous and skilled). The original dataset used in the project did not have this data, therefore we scraped the FIFA ratings from 2011 to 2016 [13]. The scraped data didn't have proper name format as in our model. Therefore, we removed the accents of the name of the player in both datasets and changed full name to "first initial. Last name". For instance, we changed "Cristiano Ronaldo" to "C. Ronaldo" in both datasets. We also kept the full name and made sure no 2 names had the same shortened names. Next, we added 2 columns to the data with the corresponding ratings of the player for the given row. The team initially decided to add these ratings and see how that impacted the accuracy of the model. But after running the model with these new predicted features, it did not impact the accuracy of the optimized architectures. The team then switched paths to explore a different angle using feature selection to understand the importance of these new predictor features. With the help of this experiment the team was able to understand the importance of these features in relation to the target variable. The ratings were imported from FIFA ratings from 2011 to 2016 [13]. This experiment would be run on all the optimized architectures and the same evaluation metrics would be used from Experiment 2.

7 Results and Discussion

In this section we will report on each experiment's results. We report the results of the balancing techniques that were used in order to make better predictions about whether a goal was scored or not. We also report the accuracies of the different architectures that were used in our experiments to determine which was the best model to use. We also used the training and prediction times to distinguish between the models since we found similar accuracies between different models. We discuss the various optimization techniques that we used for each architecture. Finally, we report our Experiment 3 which is where we added the "overall" and "shooting" ratings to see how that affected the model accuracy.

One of the balancing techniques used was oversampling. Figure 1 shows the change in the counts before and after the oversampling. We decided to test the effect that this balancing technique had on the architectures. This will be discussed below.

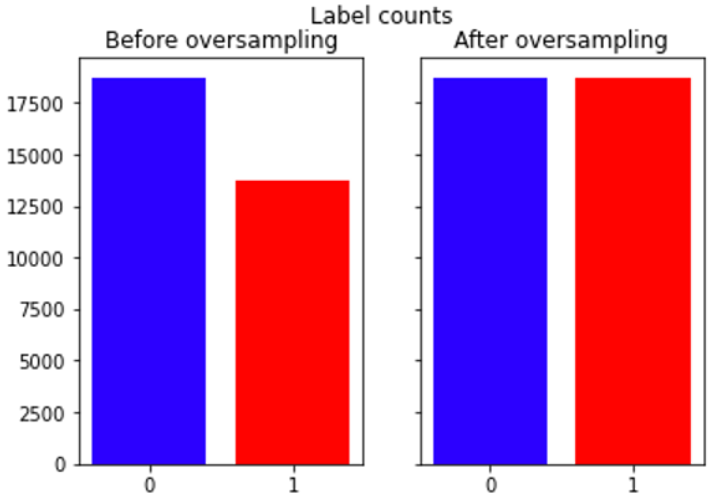

Figure 1: Target variable counts before and after balancing.

7.1 Optimization of different models

In this section we discuss how each of us optimized the different architectures used in this project. We discuss the different parameters used, and how we obtained them. The results rationale behind the model optimizations are discussed for each architecture. It should be noted that we used the SequentialFeatureSelector in sklearn on the top three features for each model in order to optimize the performance of the models in terms of efficiency. We also used this to simplify the model and remove features that would not result in significant accuracy increase.

7.1.1 Decision Tree and Random Forest

An advantage of the DTC is that we're able to graphically represent the decisions that the model has made at each step. Figure 2 shows the DTC model of the dataset before oversampling with "entropy" used to see the information gained by each decision.

The DTC and RFC were optimized using GridSearchCV. We found that the best DTC has the minimum sample leaf (min_samples_leaf) of 1, maximum depth (max_depth) of 5, and the "entropy" criterion. The tree is plotted in Figure 2. We can see from the decision tree that the feature "situation" is open play (situation = 1) is the best descriptive feature of the studied DTC. Two features "fast break" and "shot outcome is on target" are decisions being chosen to partition the training dataset. These chosen features are reasonable from a practical perspective. The hyperparameters that give RFC the best performance are with minimum sample leaf (min_samples_leaf) of 1, maximum depth (max_depth) of 3,100 estimators (n_estimators), and the "gini" criterion.

7.1.2 MLP Classifier

After running the MLP Classifier with and without balancing techniques, a subtle difference was noticed between the models. Overall, the MLP with balancing using class weight

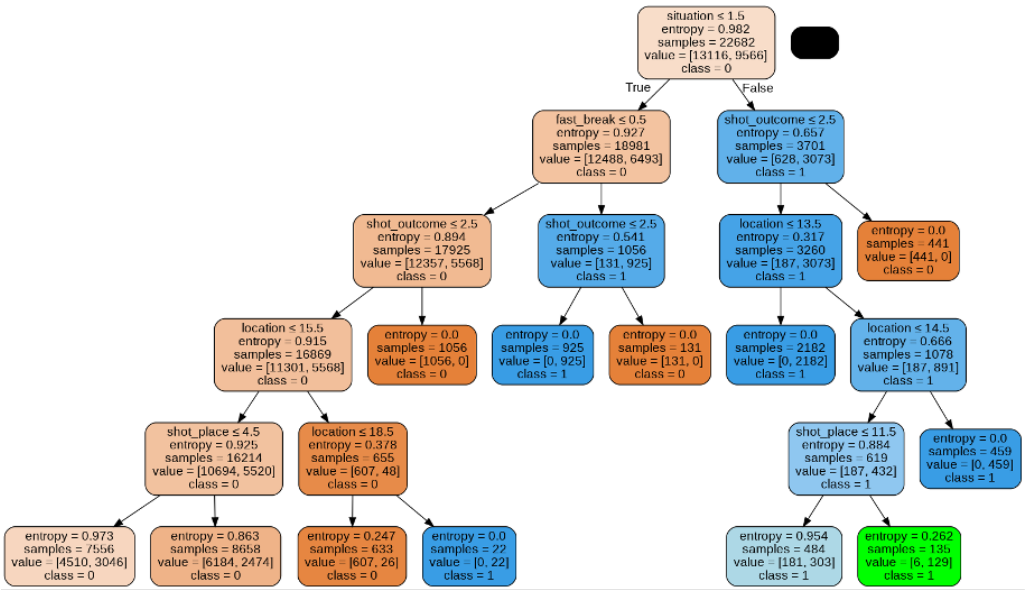

Figure 2: The tree of best DTC model on the dataset before oversampling.

techniques had a better score overall for the metrics used. Once a baseline model was developed, we did some manual hypertuning of the parameters. After many trials and tribulations, the optimized number of hidden layers was two layers with 6 nodes in each layer. Increasing the number of hidden layers or the number of nodes in each layer did not increase the scores as a saturation point was received with the following tuple (6,6,1). The results using the best parameters are reported in Table 4.

7.1.3 SVM Classifier

For the optimization of the SVM Classifier we ran GridSearchCV with values of gamma and C. The values used in GridSearchCV for $C = [0.1, 1, 10, 100]$ and $\gamma = [1, 0.1, 0.01, 0.001]$. The results of the GridSearchCV yielded the best values of $\gamma = 1$ and $C = 1$. As can be seen from the parameters used in GridSearchCV, we used the Radial Basis Function (rbf) kernel since it performed better than "linear" and "poly = 3". The values of γ and C make sense since both lower and higher values of γ and C were checked in GridSearchCV and the best parameters were in between the values for C.

7.2 Summary of Results

Table 4 displays the summary of the different models that we ran. The table also displays the different accuracy scores for the scenarios of "Without Balancing", "Oversampling", and "Class weight".

As can be seen from Table 4, the four studied models achieve accuracy of around 74% to 75%, precision of about 96%, recall score of around 42% to 43%, and F_1 score of about 59% on the dataset without balancing techniques. These performance metrics demonstrate that all models attain adequate performance. Among the four models, DTC has the best performance metrics. This can be explained that the simple structure of DTC helps it to ignore the noise of the dataset. On the oversampled dataset, the accuracy is compromised from about 75% to

Model name	**Type**	**Accuracy (%)**	**Precision (%)**	**Recall score (%)**	**F-1 Score (%)**
Random Forest Classifier	W/o Balancing	74.89	95.94	42.51	58.91
	Over Sampling	70.24	97.00	41.57	58.20
	Classweight	74.89	95.94	42.51	58.91
Decision Tree Classifier	W/o Balancing	75.05	95.97	42.89	59.29
	Over Sampling	70.15	96.31	42.12	58.61
	Classweight	**75.05**	**95.98**	**42.90**	**59.29**
MLP Classifier	W/o Balancing	74.88	95.59	42.50	58.91
	Over Sampling	70.76	96.19	42.71	59.15
	Classweight	74.88	95.94	42.50	58.91
MLP Classifier	W/o Balancing	74.89	95.94	42.50	58.91
	Over Sampling	70.74	97.12	42.72	59.34
	Classweight	74.89	95.94	42.51	58.91

Table 2: Summary of different model accuracy scores with different balancing techniques.

about 71%, and the precision is increased from around 95%-96% to around 96%-97%. This is because the oversampling technique replicates the samples that have type-1 target variables in the dataset, which makes the model after oversampling biased to the type-1 target variable. As a result, the accuracy of predicting the type-1 target variable is increased at the expense

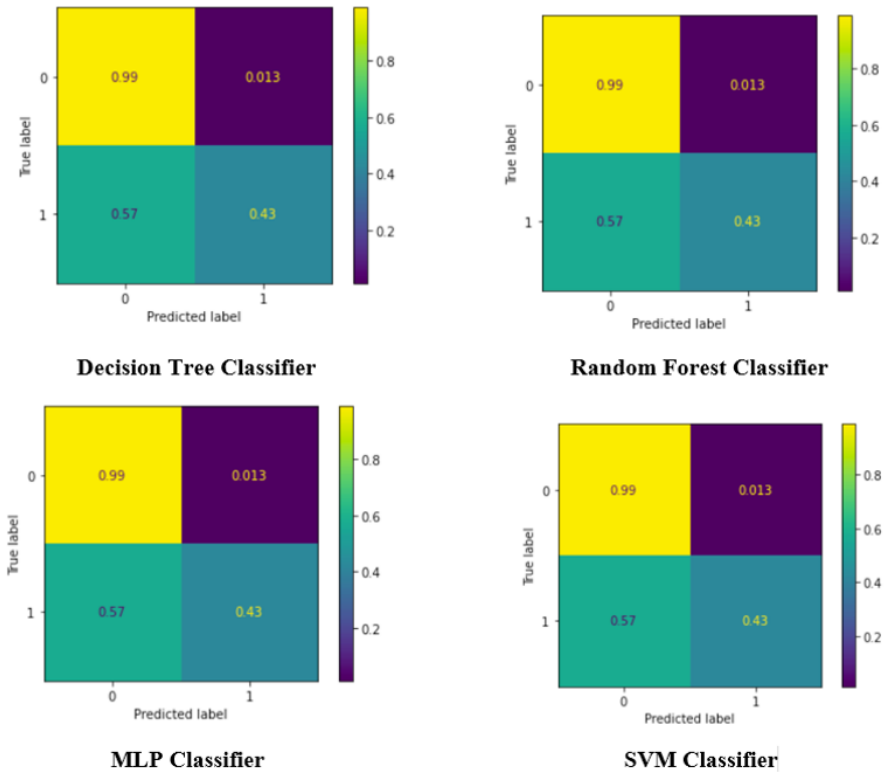

Figure 3: Confusion matrices for the best models or each architecture.

of reducing the overall accuracy. Recall and F_1 scores slightly change after oversampling the dataset. The performance metrics of all four models change negligibly after applying the class weight technique. The reason is that the class weight of the type-0 target variable and type-1 target variable is 1.3, which does not significantly affect the loss functions of the models.

The normalized confusion matrices are shown in Figure 3. What stands out is that the models perform very similarly and without looking at Table 4, we could conclude that each model performs the same. Another observation is that the model is better at predicting when an instance is not a goal as opposed to when it is a goal. Since the models performed so similarly, we decided to compute the training and prediction times to distinguish between models.

Even though we found that the DTC had the best overall performance, we decided to check whether the consideration of time would impact our decision in choosing the best model. As we can see from Figure 4, the SVM and MLP classifiers took the longest time to train and predict. However, the RFC and DTC took much lower times, both of which were under a second. This confirms our selection of the DTC as the best model in this context.

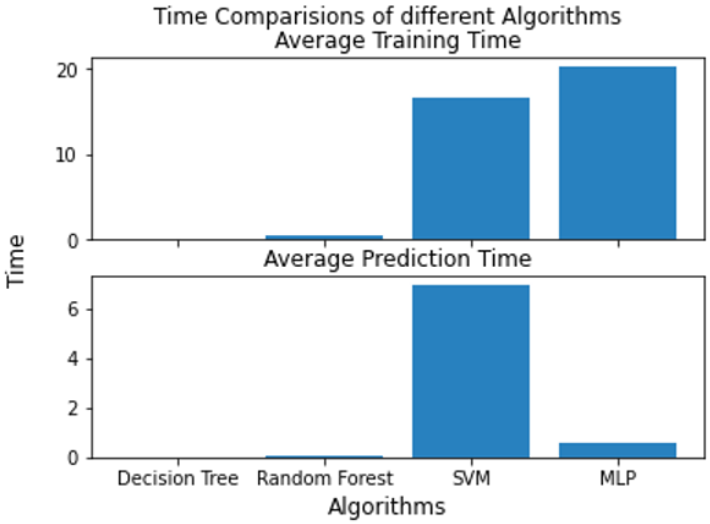

Figure 4: Training and prediction times for the DTC, RFC, SVM, and MLP architectures.

7.3 Incorporating Player Ratings

We were very hopeful about the model performance after adding the player ratings, therefore, we added the player rating features to a set of 3 best selected features for each model and ran the model bypassing the feature selection step for player rating. Surprisingly, added features didn't improve our model at all. Therefore, we went back to the feature selection step to check how important are those ratings for our prediction model. The shot rating feature was selected only if we were selecting 5 best features and overall rating was selected only if we were selecting 8 best features. This shows that rating data doesn't contribute much for the prediction purpose in our model. As expected, shot rating was more important than overall rating for the prediction.

8 Conclusions and Future work

We were able to meet the specific aims of this project. We were able to classify to some extent whether a shot would lead to a goal or not. However, the model was not good at classifying the outcome of a goal.

We found that the most important features for whether the instant resulted in a goal were “shot_outcome”, “fast_break”, “situation” for all the models except for the SVM where the top features were "shot_place", "situation" and "fast_break". This makes practical sense since these features speak to the position of the player taking a shot, the phase that the game is in, and where the player has placed the shot.

We were also successful in comparing different architectures, namely: DTC, RFC, SVM, and MLP. The comparison yielded the DTC being the best model since it has the best performance metrics, takes the shortest time to train and test, as well as being able to graphically represent the results in a way that is easier to understand and explain. In this same aim we were able to optimize each model using manual trial and error using knowledge from previous assignments for the MLP classifier and used GridSearchCV for the DTC, RFC, and SVM classifier.

Experiment 3 did not yield the results we expected. We saw that the player ratings did not affect the accuracy of the model. This is a useful finding since it means that there are more important contextual factors that affect whether a player would score a goal other than their shooting rating. This could be their positioning, how the rest of the team is playing, and some of the factors that we mention in the next paragraph.

We see from our model that, given the set of predictors we chose, it is very hard to predict the goal. It suggests that a series of events leading up to a goal is a very complex process that has a lot of variability. Time as much as milliseconds and distance as much as millimeters could make a difference between a goal or not a goal. Therefore, we might need more data in order to make such predictions. For instance, we might collect players’ and opponent players’ precise positions on the field. We might need to collect data about players’ strengths and weaknesses. For example, the player might be really good with the right foot, but may not be able to score a goal from the likely goal situation if they are forced to shoot with the left foot. Similarly, weather data (rain vs sun), venue information (home vs away), and crowd data could be incorporated into the model for better prediction. These are some of the suggestions that could improve the accuracy of our predictions.

References

[1] T. Horvat and J. Job, "The use of machine learning in sport outcome prediction: A review," *WIREs Data Min. Knowl. Discov.*, vol. 10, no. 5, p. e1380, 2020, doi: 10.1002/widm.1380.

[2] D. Berrar, P. Lopes, and W. Dubitzky,"Incorporating domain knowledge in machine learning for soccer outcome prediction," *Mach. Learn.*, vol. 108, no. 1, pp. 97–126, Jan. 2019, doi: 10.1007/s10994-018-5747-8.

[3] S. Kotsiantis, I. Zaharakis, and P. Pintelas, "Machine learning: A review of classification and combining techniques," *Artif. Intell. Rev.*, vol. 26, pp. 159–190, Nov. 2006, doi: 10.1007/s10462-007-9052-3.

[4] A. C. Constantinou, N. E. Fenton, and M. Neil, "Profiting from an inefficient association football gambling market: Prediction, risk and uncertainty using Bayesian networks," *Knowl.-Based Syst.*, vol. 50, pp. 60–86, Sep. 2013, doi: 10.1016/j.knosys.2013.05.008.

[5] R. P. Bunker and F. Thabtah, "A machine learning framework for sport result prediction," *Appl. Comput. Inform.*, vol. 15, no. 1, pp. 27–33, Jan. 2019, doi: 10.1016/j.aci.2017.09.005.

[6] Quinlan, J.R. "Induction of decision trees," *Mach Learn* 1, 81–106 (1986). https://doi.org/10.1007/BF00116251.

[7] Liaw A, Wiener M, "Classification and Regression by Random Forest.," *R News*, 2(3), 18-22, (2002).

[8] W.S. McCulloch and W. Pitts, "A Logical Calculus of Ideas Immanent in Nervous Activity," *Bull. Mathematical Bio- physics*, vol. 5, pp. 115-133 (1943).

[9] M. Minsky and S.Papert, "Perceptrons:An lntroduction to Computational Geometry," MIT Press, Cambridge, Mass., 1969.

[10] J.A. Anderson and E. Rosenfeld, "Neurocomputing: Foundations of Research," MITPress, Cambridge, Mass., 1988.

[11] Cortes, C., Vapnik, V., "Support-vector networks.," *Mach Learn*, 20, 273–297 (1995).

[12] "Football Events," https://www.kaggle.com/datasets/secareanualin/football-events (accessed Oct. 02, 2022).

[13] "Players," https://sofifa.com/players (accessed Dec. 04, 2022).