Predicting outcomes of Bundesliga matches

Timur Kulenovic, *Faculty of Computer and Information Science, University of Ljubljana*
timur.kulenovic@student.uni-lj.si

***Abstract*—Predicting sport results has always been a hot topic for numerous sport fans, no matter they had any statistical/machine learning knowledge or not. This paper tries to predict outcomes of football matches in the German Bundesliga. We tried to engineer meaningful features by obtaining data from different websites and applied our engineered dataset to various machine learning algorithms. We managed to get very promising results as ranked probability scores (RPS) value for some of our models was lower than RPS calculated with inverse odds from betting website.**

***Index Terms*—Football, Bundesliga, predicting results, machine learning, RPS, FIFA.**

I. Introduction

The main goal of this paper is to check how accurate model can we build with data that is published openly on the internet and can be scraped. Predicting outcomes of football has been very appealing not only because of academic research, but also because of economic value, since worth of football **betting market** is estimated to be almost 1 trillion dollars ([1]). Consequently, we also want to see if overcoming betting models is possible. Studies related to this paper also tried to predict number of goals scored by each team in a game, but our study will consider just predicting the winner (or draw). Therefore, we here deal with classification problem where target variable has either value **H** (home team wins), **D** (match ends up in draw) or **A** (away team wins). Because of high competitiveness of top football leagues, we do not expect extremely accurate models, The outcomes of matches also depends on many factors, which are not available or even not quantifiable.

Our work considers data from **German football league** (Bundesliga) from seasons **2011/2012 to 2018/2019**. In section III we first describe the data we obtained with web scraping. Of course, we cannot use data in the form that was obtained, so we need to think of meaningful features through the process of feature engineering described in section IV. In section V we present the results of our models. We also show why **RPS** is a good scoring rule for assessing predictions of football outcomes and compare our models to betting site model with respect to RPS rule.

II. Related work

Predicting sport results, or specifically football outcomes seems like appealing research area, so there exist a number of related works applying various machine learning and data mining methods to football data in order to produce predictive analysis.

In [1] authors used **Premier League** data spanning 11 seasons. With usage of innovative feature engineering they managed to obtain promising results, but were not able to beat the bookmaker's predictions. Authors here used the Rank Probability Score (RPS), which is argued in [2] to be the best scoring rule that accurately assesses football forecasting models.

While [1] includes also data from **video games**, [3] compares models with video games data only to those with "real" data only. With data obtained from FIFA 2015 video game they also extracted information about different strategies of teams with applying unsupervised learning methods. When comparing models built with FIFA data to those built with data from Spanish La liga they found out that accuracy of first models can get pretty close or even higher than those with real data, but on the other side they did not put much effort into feature engineering for real data. They did not use RPS for scoring rule as well.

Work in [4] also presents some other forms of features such as travel distance between teams, expert predictions, club budgets, matches with special importance and also betting odds. Even though not all papers use RPS scoring rule we observe that Gradient Boosting and Random Forest classifiers usually perform best.

III. Obtained data

In this work we are considering matches from Bundesliga, the strongest football league in Germany. We focused on seasons from 2011/2012 to 2018/2019. In table I we see the list of attributes we have scraped for each match. German football league is played in such way that each team plays against every other team twice (once at home and once away). 18 teams compete in the league (with the worst teams being relegated every season) meaning that there are 9 matches played in each of 34 rounds. Therefore, we have 306 matches for each season and in total **2448 matches** through 8 seasons. All the data describing matches was obtained with web scraping through website Soccerway.

As explained in the next section we also used data that quantifies skills of football players. From [3] we learn that models with Video games data provided promising results, so we decided to scrape the data from website Fifaindex. We obtained 34 attributes (set by football experts) for each player describing technical, physical or mental skill, where each attribute has range from 0 to 100. Attributes are split into 7 categories and are listed in II.

Additionally, from Fifaindex we also scrape following attributes for each team:

- Overall Rating
- Attack
- Midfield

TABLE I
LIST OF ATTRIBUTES FOR EACH MATCH

Attribute	Description
Season	Season of the match
Round	Round of the match
HomeTeam AwayTeam	Name of H/A team
Date Hour	Date of match Hour of match
HomeTeamPlayers AwayTeamPlayers	Players' names joined into a string
GoalsHome GoalsAway	Scored goals by H/A team
CornersHome CornersAway	Corners by H/A team
ShotsTargetHome ShotsTargetAway	Shots on target by H/A team
FoulsHome FoulsAway	Fouls commited by H/A team
OffsidesHome OffsidesAway	Offsides by H/A team
HomeTeamPossession AwayTeamPossession	% of time team had possession

- Defence
- Budget

IV. FEATURE ENGINEERING

After obtaining all the desired data, we need to go through process of feature engineering where we use our domain knowledge (of football in our case) in order to create meaningful features. Our goal in this process is to create a dataset where each example represents a match with engineered features.

Our features are based on two categories of factors:

- Team's **performance in previous matches** of the season
- Quality of team based on **experts evaluations**

A. Features based on team's performance

When quantifying team performance in the previous matches we must take into account that only matches from the current season can be considered, as there are too many factors that change between two seasons (transfers of players and coaches, clubs' budgets, additional matches in European competitions and so on). Next obstacle is that we certainly cannot create any features for the matches of the first round of the season. Furthermore, features for the games of first few rounds of season would also be quite inaccurate, so with k we will mark the number of rounds which we won't engineer features for.

First we create some simple features that measure what are the average values of team's attributes in the last *k* matches. For team's match that is played in *r*-th round and for *attr* $\in$ {ShotsOnTarget, Goals, Corners, Offsides, Possession, Fouls} we compute:

$$avg_r^{attr} = (\sum_{p=r-k}^{r-1} attr\_value_p^{attr})/k$$

where $attr\_value_p^{attr}$ is the value of team's attribute in p-th round.

That gives us 12 attributes, 6 for each team. But we do not include directly these features. As found out in [1] it is best to consider the differences of attributes for home and away teams. Therefore, selected features for match in r-th round are:

$$avgDiff^{attr} = avg\_attr_H^{attr} - avg\_attr_A^{attr}$$

where $avg\_attr_H^{attr}$ = Home Team average for attribute and $avg\_attr_H^{attr}$ = Away Team average for attribute.

Similarly, we also create a feature representing the number of points team got for their results in the last k games:

$$points_r = (\sum_{p=r-k}^{r-1} pts_p)/3k$$

where pts_p is 0 if team lost the *p*-th game, 1 if team's *p*-th ended in draw or 3 if team won the *p*-th game. Feature is normalized by dividing with 3k. Since team can get at maximum *3*k in the last 3*k* games, this means that maximum value of the feature is 1. But, again we do not include this feature, instead we include difference of Home team's points and Away team's points in the last *k* games:

$$pointsDiff = points^H - points^A$$

Work in [1] also presents a feature representing weighted points in the last k games where we sum points in the last *k* in such way that the points from the oldest game count the least and points from the previuos round (r - 1) count the most:

$$pointsWeighted_r = \sum_{p=r-k}^{r-1} \frac{(p-(r-k-1))pts_p}{3\frac{k(k+1)}{2}}$$

where $(p-(j-k-1))$ is the weight of the *p*-th match. Weight 1 is assigned to the oldest match in observation window (r - k) and weight *k* is assigned to the most recent match (r - 1). Then we normalize the feature by dividing it with sum of all weights $(k(k+1)/2)$ and the maximum number of points possible in the match (3). We select the difference ($pointsWeighted^H$ - $pointsWeighted^A$) into the set of selected features.

Next we add two features that do not take parameter *k* into consideration. First we calculate teams goal difference for r-th round:

$$GD_r = \sum_{p=1}^{r-1} S_p - \sum_{p=1}^{r-1} C_p$$

where S_p is number of scored goals in *p*-th round and C_p is number of conceded goals in *p*-th round. We then select feature representing the difference in GD:

$$GDDiff = GD^H - GD^A$$

where GD^H and GD^A are goal differences of home and away team. In the same fashion we also obtain difference in number of points for all games up to r-th round:

$$pointsTotalDiff = pointsTotal^H - pointsTotal^A$$

TABLE II
LIST OF ATTRIBUTES FOR EACH MATCH

Ball Skills	Defence	Mental	Passing	Physical	Shooting	Goalkeeper
Ball Control	Marking	Aggression	Crossing	Acceleration	Heading	GK Positioning
Dribbling	Slide Tackle	Reactions	Short Pass	Stamina	Shot Power	GK Diving
	Stand Tackle	Attacking Position	Long Pass	Strength	Finishing	GK Handling
		Interceptions		Balance	Long Shots	GK Kicking
		Vision		Sprint Speed	Curve	GK Reflexes
		Composure		Agility	Free Kick Accuracy	
				Jumping	Volleys	

where $pointsTotal^H$ and $pointsTotal^A$ represent sum of points in the season (up to r-th round) for home and away team.

For now we described some meaningful features, that take into account team's performance in the last *k* games, but did not really consider who were team's opponents in these last *k* games. Winning all previous games surely tells more of the quality of team's performance when the opponents were high performing teams as oppose to the case when opponents were low performing teams. Authors in [1] also present value called Form that is updated for a team after every match. Each team starts the season with value 1, then it's form is updated in the following way:

- If Home team wins:

$$Form_r^H = Form_{r-1}^H + \alpha Form_{r-1}^A$$
$$Form_r^A = Form_{r-1}^A - \alpha Form_{r-1}^A$$

- If Away team wins:

$$Form_r^A = Form_{r-1}^A + \alpha Form_{r-1}^H$$
$$Form_r^H = Form_{r-1}^H - \alpha Form_{r-1}^H$$

- If match ends in draw:

$$Form_r^H = Form_{r-1}^H - \alpha(Form_{r-1}^H - Form_{r-1}^A)$$
$$Form_r^A = Form_{r-1}^A - \alpha(Form_{r-1}^A - Form_{r-1}^H)$$

When home teams wins, it steals the fraction (α) of away team's form and adds it to his own form. When team beats low performing team it's form will grow in smaller extent than it would grow in case of beating high performing team.

If draw happens, update is constructed in such way that better team would lose form and worse team would gain some form.

After each completed round the sum of all teams' forms should equal to 18, as there 18 teams in the league.

We tried different values for α and find out that with $\alpha = 0.33$ as in [1] we can most of the time build models with better results than with other values α. In any case, we add $FormDiff = Form_r^H - Form_r^A$ to list of features.

Of course, from the *GoalsHome* and *GoalsAway* attributes of the match we also get the **target feature** *Result* that has value:

- H, if home team scored more goals
- D, if teams scored equal number of goals
- A, if away team scored more goals

B. Features based on experts evaluations

First 5 features based on data from Fifaindex website are differences in home and away team's basic attributes:

$$RatingDiff_{attr} = Rating_{attr}^H - Rating_{attr}^A$$

where $attr \in \{$Attack, Defence, Midfield, Overall Rating and Budget$\}$.

Next, we proceed with adding features based on individual players qualities. Here we place a cautionary advice that in order to create following features we must know which players were playing in the match. This is not a problem when building a model based on previous data, but it becomes an obstacle when we want to predict the outcome of the match that happens tomorrow. We can create a set of features based on team's performance in the past games but at that point we cannot create features described in this subsection. However, teams are required to publish list of starters an hour before the match, so we can still create a full set of features within an hour before the match starts.

Also, note that we will create features based on 11 starting players that we scraped data for, we did not consider any players that got into the game later as substitutions.

We begin with creating 14 features based on physical and mental skills of players. For each skill that falls into category of either physical or mental skills (shown in figure II we compute the average of 10 players (we exclude goalkeepers as we judge that their mental and physical skills do not affect the game). Selected features are then differences of attribute averages for each attribute:

$$PMDiff_{attr} = PMavg_{attr}^H - PMavg_{attr}^A$$

where $attr \in Physical \cup Mental$ and $PMavg_{attr} = (\sum_{i=1}^{i=10} PM_i)/10$

Next, we see that we have two attributes evaluating players' ability to tackle (Slide Tackle and Stand tackle). We calculate the average of those two for each player in the team and then select top 5 tacklers in the team (it's not crucial for the team that all players are good tacklers, we decided to select 5). We get the average of top 5 tacklers and denote it as *TC*. Then we find team's top 5 players for each of two Ball Skills attributes (Ball Control and Dribling) and compute averages for these two attributes (*BC* and *DR*). These two attributes are lowered by the values of opponent's *TC*. We add following features into our set:

$$BCtrlDiff = (BC^H - TC^A) - (BC^A - TC^H)$$
$$DribbleDiff = (DR^H - TC^A) - (DR^A - TC^H)$$

Similarly as in the previous paragraph we create features for Passing skills. From each passing skill (Crossing, Short Pass, Long Pass) the value of opponent's average value for Marking attribute is subtracted. We denote average value of top 5 players who are the best at Crossing as *CR*, same for Short Pass as *SP* and for Long Pass as *LP*. We also get team's average of team's five players who are best at Marking, we denote it as *MA*. Features we create are:

$$CrossingDiff = (CR^H - MA^A) - (CR^A - MA^H)$$

$$ShortPassDiff = (SP^H - MA^A) - (SP^A - MA^H)$$

$$LongPassDiff = (LP^H - MA^A) - (LP^A - MA^H)$$

Finally, we compute the average values of team's goalkeeper goalkeeping skills and denote the quality of goalkeeper as *GK*. Then we compute values of team's Shooting skills by averaging values of all players for each Shooting attribute. We name this averages *HE*, *ShP*, *FIN*, *FK*, *CU*, *PEN*, *VO* for attributes Heading, Shot Power, Finishing, Long Shots, Curve, FK Acc., Penalties, Volleys, respectively. We lower this attributes by value of opponent's *GK*. Our last 7 engineered features are:

$$HeadingDiff = (HE^H - GK^A) - (HE^A - GK^H)$$

$$ShotPowerDiff = (ShP^H - GK^A) - (ShP^A - GK^H)$$

$$FinishingDiff = (FIN^H - GK^A) - (FIN^A - GK^H)$$

$$FKAccDiff = (FK^H - GK^A) - (FK^A - GK^H)$$

$$CurveDiff = (CU^H - GK^A) - (CU^A - GK^H)$$

$$PenaltiesDiff = (PEN^H - GK^A) - (PEN^A - GK^H)$$

$$VolleysDiff = (VO^H - GK^A) - (VO^A - GK^H)$$

Since we produced quite a number of different features, we present an overview of them in III.

TABLE III
LIST OF ALL ENGINEERED FEATURES

Feature	Additional description
ShotsOnTargetAvgDiff	Diff. of avg of Sh. on target in last k games
GoalsAvgDiff	Diff. of avg of goals in last k games
CornersAvgDiff	Diff. of avg of corners in last k games
OffsidesAvgDiff	Diff. of avg of offsides in last k games
PossessionAvgDiff	Diff. of avg of possession in last k games
FoulsAvgDiff	Diff. of avg of fouls in last k games
PointsDiff	Diff of normalized pts. in last k games
PointsWghtDiff	Diff. of norm. and weight. pts. in last k games
GDDiff	Diff. of Goal Differences in all games played
PointsTotalDiff	Diff. of points in all games played
FormDiff	Diff. of forms
AttackDiff	Diff. of attack ratings
DefenceDiff	Diff. of defence ratings
MidfieldDiff	Diff. of midfield ratings
OverallDiff	Diff. of overall ratings
BudgetDiff	Diff. of budgets
AccelerationDiff	Diff. of avgs of acceleration
StaminaDiff	Diff. of avgs of stamina
StrengthDiff	Diff. of avgs of strength
BalanceDiff	Diff. of avgs of nalance
SprintSpeedDiff	Diff. of avgs of sprint speed
AgilityDiff	Diff. of avgs of agility
JumpingDiff	Diff. of avgs of jumping
AggressionDiff	Diff. of avgs of aggression
ReactionsDiff	Diff. of avgs of reactions
AttPositionDiff	Diff. of avgs of attack position
InterceptionsDiff	Diff. of avgs of interceptions
VisionDiff	Diff. of avgs of vision
BCtrlDiff	Diff. of ball control lowered by tackle value
DribbleDiff	Diff. of dribble lowered by tackle value
CrossingDiff	Diff. of crossing lowered by marking value
ShortPassDiff	Diff. of short pass lowered by marking value
LongPassDiff	Diff. of long pass lowered by marking value
HeadingDiff	Diff. of heading lowered by GK value
ShotPowerDiff	Diff. of shot power lowered by GK value
FinishingDiff	Diff. of finishing lowered by GK value
FKAccDiff	Diff. of free kick values lowered by GK value
CurveDiff	Diff. of curve lowered by GK value
PenaltiesDiff	Diff. of penalties lowered by GK value
VolleysDiff	Diff. of volleys lowered by GK value

V. RESULTS

To get the optimal value of k parameter we tried different values and it turned out that $k = 5$ works well with our dataset, meaning that we will not engineer features for the first 5 rounds for each season. To evaluate the models we did not use cross validation since that would mean scoring on the model that was trained on not yet existent data. Therefore, we applied different algorithms to our data, where **seasons from 2011/2012 to 2017/2018 was our training data and season 2018/2019 was our test data**.

In tables IV and V we see confusion matrix and precision-recall-f1 values for **logistic regression**. We see that logistic regression has high recall value for Home win in comparison to recall values for Away and Draw. We see that it very rarely predicts Draw outcome, even though there isn't so few actual draws.

Next, we applied our data to **random forest classifier**. Since this classifier return scores that are dependent on random value we build and ran the classifier 50 times and considered the average values. From table VI we see the average precision and recall values for random forest. By considering these values it does slightly better job than logistic regression, but numbers are quite similar.

TABLE IV
LOGISTIC REGRESSION CONFUSION MATRIX.

	Pred. Home	Pred. Draw	Pred Away
Actual Home	97	0	20
Actual Draw	41	3	16
Actual Away	31	2	51

We also tried **gradient boosting** algorithm, that proved as the most accurate in [1]. From we see average values for precision, recall and F1 score in table VII we observe that it does better job classifying draw outcomes than previous two algorithms.

TABLE V
LOGISTIC REGRESSION PRECISION-RECALL-F1 TABLE

	Precision	Recall	F1-Score
Home	0.574	0.829	0.678
Draw	0.6	0.05	0.092
Away	0.586	0.607	0.596

TABLE VI
RANDOM FOREST CLASSIFIER PRECISION-RECALL-F1 TABLE.

	Precision	Recall	F1-Score
Home	0.5850	0.8643	0.6977
Draw	0.71	0.0297	0.0566
Away	0.5909	0.6005	0.5956

TABLE VII
GRADIENT BOOSTING PRECISION-RECALL-F1 TABLE.

	Precision	Recall	F1-Score
Home	0.6215	0.8352	0.7127
Draw	0.4231	0.15	0.2215
Away	0.5999	0.5890	0.5944

Lastly, we use data to build **multilayer perceptron classifier**. We present it's average precision-recal-f1 values in table VIII. Once more, low recall value for draw tells us that classifier rarely predicts draw as outcome.

TABLE VIII
MULTILAYER PERCEPTRON CLASSIFIER PRECISION-RECALL TABLE.

	Precision	Recall	F1-Score
Home	0.5667	0.8492	0.6799
Draw	0.9067	0.0313	0.6042
Away	0.5890	0.5852	0.5867

In IX we present the accuracy values for each of the four algorithms (for the last 3 we show the averages). Like in the precision-recall-f1 tables, we see here that there are no major differences in accuracy.

TABLE IX
ACCURACY OF MODELS

Model	Accuracy
Logistic Regression	0.5785
Random Forest	0.5875
Gradient Boosting	0.5985
MLP Classifier	0.5762

A. RPS scoring rule

While confusion matrices, accuracy, precision, recall and f1-score give us some information about our models, there are scoring rules that assess models better. In [2] it was shown through analysis that **ranked probability score** (RPS) evaluates football forecasting models better than other scoring rules like the Brier score and maximum log likelihood. Introduced by Epstein in 1969 [5] RPS basically measures how well does a probability distribution of our predictions match the observed outcome. By definition from [2]:

$$RPS = \frac{1}{r-1}\sum_{i=1}^{r}(\sum_{j=1}^{i}p_j - \sum_{j=1}^{i}e_j)^2$$

where r is the number of outcomes (in our case 3) and p_j and e_j are prediction and true outcomes at position j. For example, if we predicted for certain match that home team wins with 0.7 probability, that draw will happen with 0.25 probability and that away team wins with 0.05 probability, then $p = [0.7, 0.25, 0.05]$. If draw actually happened, then the vector of true outcome would simply be $e = [0, 1, 0]$. What makes RPS an adequate scoring rule is that it takes into account that we are dealing with ordinal values and predicting draw when home team wins isn't as bad as predicting away team's victory. Since we want the difference of both distributions to be low, smaller RPS means better performance of a model. We get the RPS of our model by simply calculating the average of RPS of all matches.

We wanted to compare our results to model from betting site presented in the form of inverse odds. We scraped the inverse odds for all Bundesliga matches in the 2018/2019 season from the website Betexplorer. Let the inverse odds for i-th game be:

$$O = [o_H, o_D, o_A]$$

To get probabilities we first need to get implied probabilities (π). Let implied probabilities for i-th game be:

$$\pi = [\frac{1}{o_H}, \frac{1}{o_D}, \frac{1}{o_A}] = [\pi_H, \pi_D, \pi_A]$$

Since bookmakers want bigger profit these implied probabilities do not sum up to 1, but usually from 1.05 to 1.07. So, to get actual probabilities we need to divide the implied probabilities with the sum of implied probabilities:

$$p = \frac{[\pi_H, \pi_D, \pi_A]}{\sum_{m=H,D,A}\pi_m} = [p_H, p_D, p_A]$$

These are probabilities used in calculating the RPS for Betexplorer. Since our models considered only games from round 6 onwards, we did the same when calculating Betexplorer RPS. The comparison is visible in table X.

TABLE X
RPS VALUES

Model	Value
Betexplorer	0.1931
Logistic Regression	0.1904
Random Forest	0.1912
Gradient Boosting	0.1918
MLP Classifier	0.1900

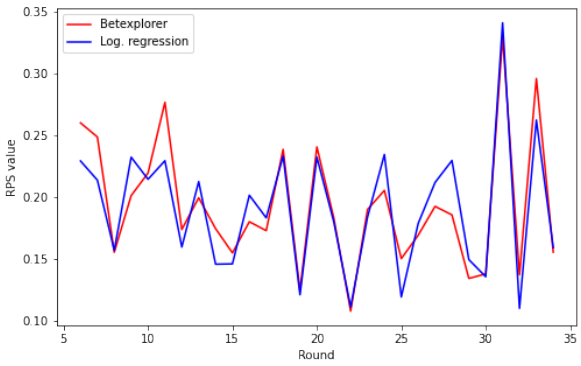

Fig. 1. **RPS comparison for each round.** We see for each round whether our logistic regression model or Betexplorer had higher RPS.

Since the RPS values for random forest, gradient boosting and MLP are averages of building models 50 times, we also present histograms of RPS values for each of these algorithms in figure 2. While MLP indeed has the lowest RPS values, values here also have the highest variance. On the other side, gradient boosting has the highest RPS values, but are not so dispersed. In figure 1 we show how logistic regression performed in each round compared to RPS from Betexplorer.

VI. Conclusion

This paper concludes that with adequate engineering and selection of attributes it is possible to reach the quality and accuracy of bookmakers' prediction models. Our work only considers one season of specific football league, so extensive conclusions cannot be made, but results obtained are definitely promising as all four algorithms used in this paper were able to score RPS values lower than the one obtained from betting website. First thing to be done in further analysis is to check how accurate is our model when predicting scores of other seasons or even other competitions. We could also use featured engineered to build regression models to predict number of goals, scored by each team. There is also much room in improving the quality of existing attributes and also in creating new attributes, we could for example also track the form of the players and not use the static values from evaluated by experts. We conclude that although we showed that it might possible to gain a profit from betting with the help of machine learning methods, there are still many possibilities to further reduce the RPS value of models.

References

[1] R. Baboota and H. Kaur, "Predictive analysis and modelling football results using machine learning approach for english premier league," *International Journal of Forecasting*, vol. 35, no. 2, pp. 741–755, 2019.
[2] A. C. Constantinou and N. E. Fenton, "Solving the problem of inadequate scoring rules for assessing probabilistic football forecast models," *Journal of Quantitative Analysis in Sports*, vol. 8, no. 1, 2012.
[3] J. Shin and R. Gasparyan, "A novel way to soccer match prediction," *Stanford University: Department of Computer Science*, 2014.
[4] N. Tax and Y. Joustra, "Predicting the dutch football competition using public data: A machine learning approach," *Trans. Knowl. Data Eng*, vol. 10, no. 10, pp. 1–13, 2015.
[5] E. S. Epstein, "A scoring system for probability forecasts of ranked categories," *Journal of Applied Meteorology (1962-1982)*, vol. 8, no. 6, pp. 985–987, 1969.

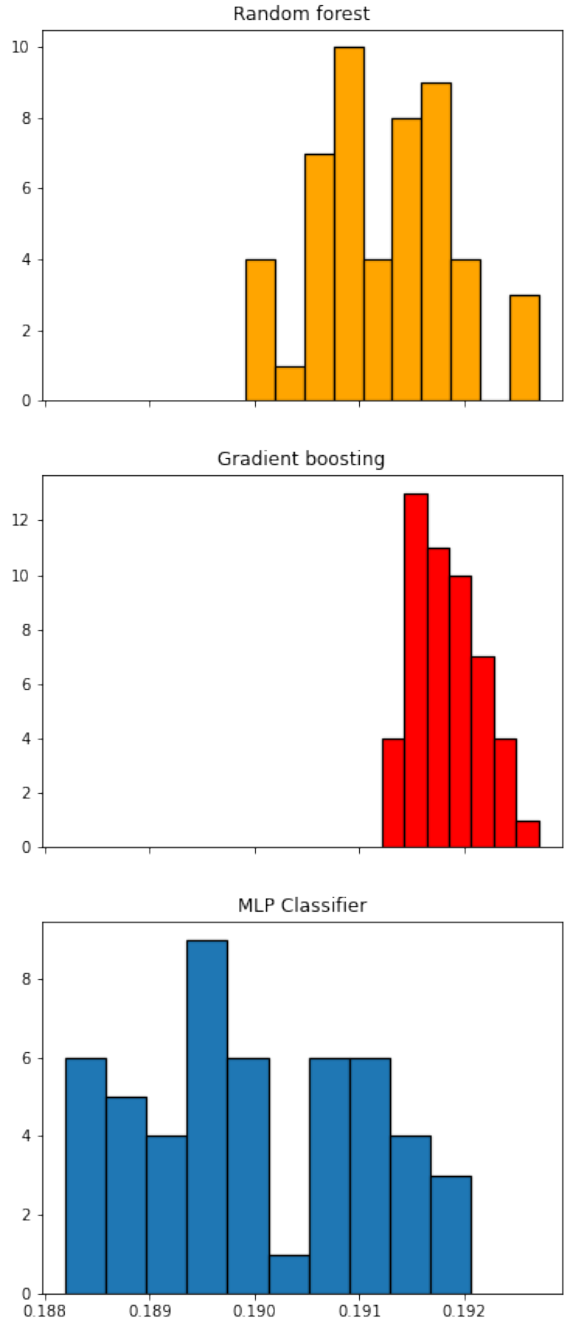

Fig. 2. **Histogram of RPS values for algorithms.** X axis is shared in order to compare values easier.