Football Match Outcome Prediction using DeepSet Player Aggregation

Group 12

Sayed Masoud Mousavi - 9735453
Mohammad Bahrami - 9724133

June 5, 2022

Contents

1 Problem Understanding Phase

1.1 Problem Definition and Scope

Association Football is the world's most popular sport. Each match is played between 2 teams of 11 players each. Teams try to score goals and the team scoring the more goals wins. If both teams have the same number of goals scored, the result will be a tie. The better the players of a team, the higher the chances of that team winning a match. In every match, there are events of player actions, either positive including but not limited to scoring a goal, successfully passing the ball and intercepting the ball, or negative, for instance scoring an own goal, getting a red or yellow card and etc. In addition to players' merit and quality playing, other factors such as match context, weather conditions, coaching team and players' experience, to name a few, as well as pure luck play an enormous role in determining the outcome of a match.

Each match is held at a stadium, usually filled with cheering audience applauding their favorite team. The team playing at their home stadium have an advantage since the audience will have a supporting effect and the boost in morale of the home team players will cause the home team to have a higher chance of winning.

Due to the sport's nature, predicting the outcome of a match is rather difficult. Moreover, in contrast to traditional ML problems where the output is a function of an input vector, the outcome of a match is a function of two input vectors, the home and the away feature vectors and if the players' features are to be modeled as well, there will be two sets of features for the home and the away teams. Since each team is represented as a set of players and sets are permutation-invariant, any modeling will also have to be permutation invariant.

1.2 Previous Works

Prior approaches to predict the outcome of a match are mostly statistical methods originating from zero-sum games such as chess prediction.

1.2.1 Elo Rating System

Elo rating system is one popular method that tries to calculate the relative skill level of the two teams competing in a match. A set of fixed mathematical formulas are introduced in this approach which aim to rate teams based on their previous performances. If a team has won more matches and against higher-skilled teams, their rating is higher. The ELO rating system's formulas have a number of hyper-parameters that could be tuned for better results. Initially, the teams are given a initial score. For each match, the score of participating teams of that match is updated utilizing equation 3.

To Update the Elo scores, the expected result for the home and away teams are calculated based on their current scores. The expected results for the home and away teams are also used to predict the result of a match

$$E_t^H = \frac{1}{1 + c^{(R_t^A - R_t^H)/d}} \tag{1}$$

$$E_t^A = 1 - E_t^H = \frac{1}{1 + c^{(R_t^H - R_t^A)/d}} \tag{2}$$

Where E_t^H and E_t^A are the expected result for the home and away teams at time t respectively. R_t^H and R_t^A are the ratings of home and away team at time t respectively and c and d are meta-parameters set according to the world football ratings standards.

After calculating the expected results of participating teams, their Elo scores can be updated based on the expected results and the final outcome of the match.

$$R_{t+1}^H = R_t^H + K(O_t^H - E_t^H) \tag{3}$$

where R_{t+1}^H is the rating of the home team at time $t+1$, R_t^H is the rating of the home team at time t, K is a meta-parameter set by the world football ratings standards and can be interpreted as a learning rate, O_t^H is the actual outcome of the match at time t for the home team which is set to 1 for a victory, 0.5 for a draw and 0 for a defeat and E_t^H is the expected outcome of the match for the home team at time t formulated in Eq. 1.

The rating for the away team is updated with the same rule and with respect to away team's previous rating and expected and actual outcome of the match. The predictions with this method is done at the time of calculating the expected scores. for the teams involved in a match, as the following terms:

$$Prediction = \begin{cases} \text{Draw} & \text{if } \left|E_t^H - E_t^A\right| \leq T \\ \text{Home Victory} & \text{if } E_t^H - E_t^A > T \\ \text{Home Defeat} & \text{if } E_t^H - E_t^A < T \end{cases} \tag{4}$$

where E_t^H and E_t^A are expected values of home and away and calculated from Eq. 1, 2 respectively and T is a hyperparameter called the draw threshold and tuned with the validation set and chosen from the values $0.01, 0.03, 0.1$ and 0.3 for each league.

1.2.2 Blade Chest

Other approaches combine both ML methods and statistical formulas utilizing both hidden representations and fixed mathematical relations.

Blade Chest model is one example of this approach. Each team has a vector of features obtained prior to a match. Two feed forward encoders are used to attain two hidden vectors for a team. One being the blade vector representing the offensive strategy and strength of each team and the other being the chest vector representing the defensive strategy and strength. Using a mathematical operation as the decoder, the match-up score is calculated as:

$$S = Home_{\text{blade}} \cdot Away_{\text{chest}} - Away_{\text{blade}} \cdot Home_{\text{chest}} \tag{5}$$

Using thresholds, this score can be binned into regions where higher scores correspond to the winning territory for home team and lower scores correspond to the winning territory for away team and a middle ground corresponds to ties.

1.2.3 Simple GNN

Recently and with the rising popularity of graph models, a graph based model has been used to create a message-passing network of team nodes. This model is transductive and each team has a node in the network and the edges are of types win and lose with higher edge weights corresponding to more recent matches. This weighting of edges is through fixed mathematical formulas and aims at favoring more recent match links.

1.3 Our Ideas

Our approach aims to represent each team competing in a match as a multi-set of 11 players comprising the lineup of that team. The problem will be a classification task whose target variable will be a vector of 3 possible outcomes of a match, home win, home loss and tie. The target variable will be a function of two multi-sets home and away, each containing 11 players.

Players have two sets of measures, post-game and pre-game. Pre-game measures are attributes such as market value, age and video game rating, conducted by video game companies such as EA and Konami. Post-game measures such as goals scored or minutes played are not accessible prior to the match.

Post-game measures contain a tremendous amount of information in regard to individual players. Our idea is to utilize these information through pre-game aggregates of these post-game measures. In other words, for each player, a new set of aggregate pre-game attributes are derived from the post-game measures of the last n games, n can be tuned. As a concrete example, each player's total number of successful passes in the last 3 matches is an aggregate pre-game attribute.

One key note to keep in mind is that the nature of this task is both dependent and independent of order. At intra-team level, there is no fixed point of reference and hence each team is represented as a multi-set and any subsequent modeling need to preserve this permutation invariance. In contrast, at inter-team level, for the home team advantage, order is meaningful and the home team's position need be fixed at the time of modeling.

As an increment to the universal approximation theorem, the injective multi-set theorem is utilized to map an injective function from the set of players to their team. This approach will be used to represent each team as a multi-set of its players.

Each team's multi-set is turned into a vector injectively with the following formula:

$$V_T = \text{MLP}_\theta(\sum_{p \in T} \text{MLP}_\phi(p)) \tag{6}$$

Where V_T is the final aggregated vector for team T and p is the player feature vector for all the players of team T.

Each MLP is a neural fully connected network with at least one hidden layer. After obtaining each team's hidden representation, the vector of both teams can be fed into another MLP whose output layer is the target variable of the match outcome

2 Data Preparation

2.1 Outlier Detection

Field players and goalkeepers are the input to our problem. 3 attributes for field players including goals scored, market value and video game rating plus 1 attribute of saves for goalkeepers are used for this section.

2.1.1 Z score

	postGame_goals
37467	2
50688	2
50757	2
50877	2
51107	2
...	...
67259	4
47229	4
27278	4
26430	4
18678	5

[944 rows x 1 columns]

(a) player post game goals

	preGame_marketValueMilEuro
36	51.5
41756	51.5
41713	51.5
18798	51.5
41603	51.5
...	...
65458	525.0
67300	525.0
69766	525.0
69134	525.0
65649	525.0

[1581 rows x 1 columns]

(b) player pre game market value

	preGame_overall
66768	48.0
64234	50.0
51624	50.0
58194	50.0
53924	50.0
...	...
42746	94.0
42882	94.0
43773	94.0
41757	94.0
43515	94.0

[312 rows x 1 columns]

(c) player pre game video game score

	postGame_save
4385	10.0
3748	10.0
4517	10.0
4626	10.0
4802	10.0
4808	10.0
5141	10.0
5198	10.0
5424	10.0
6029	10.0
6176	10.0
6810	10.0
7027	10.0
7088	10.0
7120	10.0

(d) goalkeeper post game saves

Figure 1: Outliers in the four selected fields with the Z score method

First, we perform a Z-score transformation on the data. Then every player that has a value greater than 3 or smaller than -3 in a specific field is considered to be a outlier. then we check the values to see if there is anything wrong with the data. which in our case, for example for the post game goals, most players do not score any goals in a specific match. This causes the other player who score goals to be recognized as outliers. Some of the outliers and their count is shown in figure 1.

2.1.2 IQR

	postGame_goals
39708	1
51438	1
51409	1
51407	1
51406	1
...	...
27278	4
47229	4
49567	4
67259	4
18678	5

[8211 rows x 1 columns]

(a) player post game goals

	preGame_marketValueMilEuro
44571	24.0
54333	24.0
6470	24.0
54515	24.0
28821	24.0
...	...
67300	525.0
69313	525.0
67346	525.0
69766	525.0
65458	525.0

[6949 rows x 1 columns]

(b) player pre game market value

	preGame_overall
66768	48.0
58194	50.0
64234	50.0
51624	50.0
62214	50.0
...	...
44999	94.0
38435	94.0
44946	94.0
28548	94.0
36068	94.0

[532 rows x 1 columns]

(c) player pre game video game score

	postGame_save
7956	8.0
4268	8.0
6609	8.0
4410	8.0
4498	8.0
...	...
1877	13.0
6168	13.0
6971	14.0
6526	16.0
4798	17.0

[232 rows x 1 columns]

(d) goalkeeper post game saves

Figure 2: Outliers in the four selected fields with the IQR method

We perform the standard IQR method and we can see the values considered as outliers with this method in figure 2. The key point to note here is that this method seems to be more sensitive to outliers in our data as the number of outliers found is much greater comparing with the z-score method.

2.2 Transformation and Standardization

```
preGame_overall
Original data skewness: 0.11
After transformation if necessary data skewness: -0.053
----------------------------------------
preGame_potential
Original data skewness: 0.094
After transformation if necessary data skewness: -0.007
----------------------------------------
preGame_marketValueMilEuro
Original data skewness: 9.837
After transformation if necessary data skewness: 0.104
----------------------------------------
preGame_ageDays
Original data skewness: 0.28
After transformation if necessary data skewness: 0.067
----------------------------------------
```

(a) players' selected fields

```
preGame_overall
Original data skewness: 0.079
After transformation if necessary data skewness: -0.042
----------------------------------------
preGame_potential
Original data skewness: 0.101
After transformation if necessary data skewness: -0.004
----------------------------------------
preGame_marketValueMilEuro
Original data skewness: 3.05
After transformation if necessary data skewness: 0.173
----------------------------------------
preGame_ageDays
Original data skewness: 0.219
After transformation if necessary data skewness: 0.022
```

(b) goalkeepers' selected fields

Figure 3: Skewness of selected fields before and after choosing the best transform

Numerical non-binary fields have been chosen on which 6 candidate transforms including 3 methods of log, square root and inverse square root, each with 2 added values of 1 and 0.1, were performed and the approach with the least skewness were selected and finally all values were normalized with the z-score standardization method. Since log and inverse square root cannot accept negative or zero values, all data are added with their respective minimum and then again with either 1 or 0.1.

2.3 Reclassifying the Categorical Features

	preGame_position	preGame_rc_position
0	DC	D
1	FW	F
2	MC	M
3	MR	M
4	MC	M
...	...	...
80075	AMC	M
80076	DMC	M
80077	FW	F
80078	FW	F
80079	AMC	M

Figure 4: the pre game positions before and after reclassifying

Non-numeric categorical attributes are outnumbered by numerical attributes and are limited to 'preGame_side', 'preGame_line', 'preGame_position', 'preGame_preferredFoot', of which only 'preGame_position' could be reclassified. As a concrete example, 'FW','FWL' and 'FWR' could all be reclassified as 'F', short for 'Forward'.

2.4 Binning

```
numerical_fp = ['postGame_minPlayed',
  'preGame_overall', 'preGame_potential', 'preGame_marketValueMilEuro',
  'preGame_ageDays', 'postGame_error',
  'postGame_clearance', 'postGame_index', 'postGame_shots',
  'postGame_shots_on_target', 'postGame_shots_left_foot',
  'postGame_shots_right_foot', 'postGame_shots_head',
  'postGame_shots_other', 'postGame_goals', 'postGame_goals_left_foot',
  'postGame_goals_right_foot', 'postGame_goals_head',
  'postGame_goals_other', 'postGame_xG', 'postGame_cross',
  'postGame_cross_success', 'postGame_pass', 'postGame_pass_success',
  'postGame_pass_final_third', 'postGame_pass_final_third_success',
  'postGame_pass_forward', 'postGame_pass_forward_success',
  'postGame_dribble', 'postGame_dribble_success', 'postGame_tackle',
  'postGame_tackle_success', 'postGame_interception',
  'postGame_challenge', 'postGame_ball_recovery', 'postGame_ball_lost',
  'postGame_key_pass', 'preGame_xgpm', 'preGame_xppm',]

numerical_gk = ['postGame_minPlayed',
      'preGame_overall',
       'preGame_potential', 'preGame_marketValueMilEuro',
       'preGame_ageDays', 'preGame_xgpm',
       'preGame_xppm', 'postGame_error', 'postGame_clearance',
       'postGame_index', 'postGame_pickUp', 'postGame_punch', 'postGame_save']
```

Figure 5: All the binned features for the field players and goalkeepers

For each of the players, 2 methods of binning with equal width and equal frequency, each with 4 random bins between 3 and 9, have been used and the new binned values have been saved.

	postGame_pass	postGame_pass_binned_6
0	52	(32.667, 65.333]
1	16	(-0.196, 32.667]
2	51	(32.667, 65.333]
3	46	(32.667, 65.333]
4	98	(65.333, 98.0]
...	...	...
80075	14	(-0.196, 32.667]
80076	9	(-0.196, 32.667]
80077	22	(-0.196, 32.667]
80078	8	(-0.196, 32.667]
80079	10	(-0.196, 32.667]

(a) binned with equal width with 6 bins

	postGame_pass	postGame_pass_binned_4_ef
0	52	(44.0, 196.0]
1	16	(-0.001, 21.0]
2	51	(44.0, 196.0]
3	46	(44.0, 196.0]
4	98	(44.0, 196.0]
...	...	...
80075	14	(-0.001, 21.0]
80076	9	(-0.001, 21.0]
80077	22	(21.0, 31.0]
80078	8	(-0.001, 21.0]
80079	10	(-0.001, 21.0]

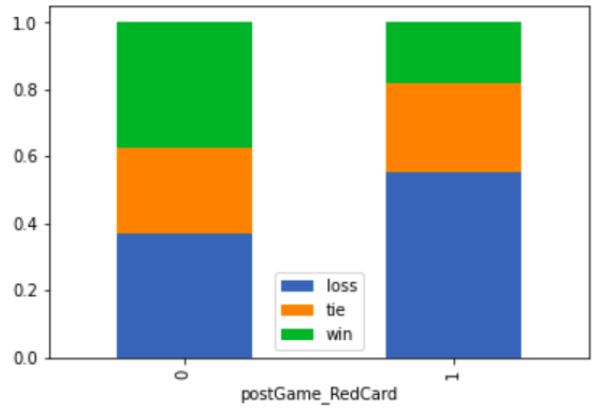
80080 rows × 2 columns

(b) binned with equal frequency with 4 bins

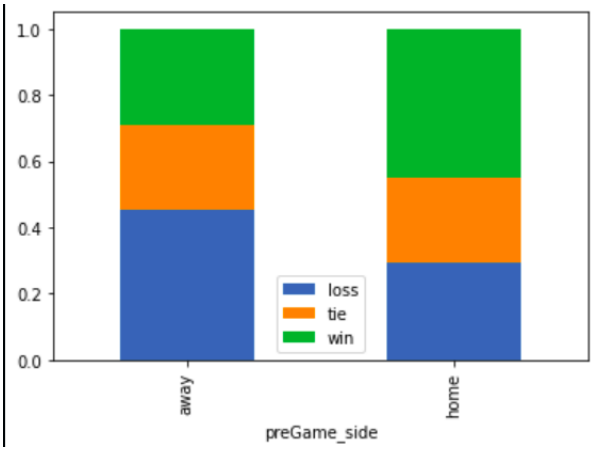
Figure 6: Players post game passes binned with two different methods and different number of bins

3 Exploratory Data Analysis

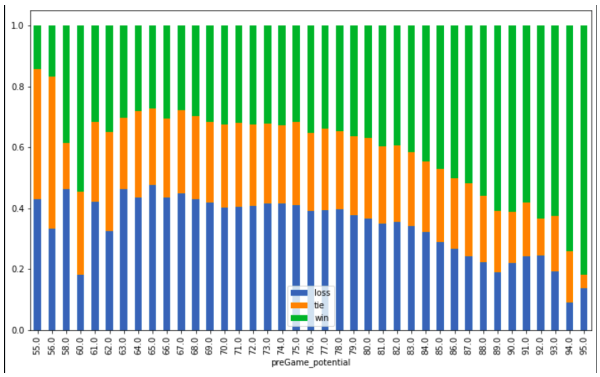
3.1 Univariate Relations with the Target Value



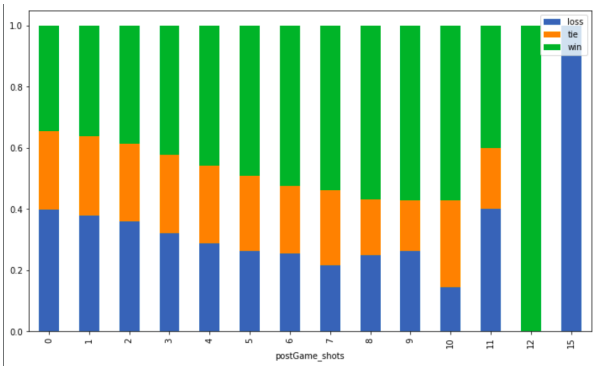
(a) weather a player received a red card or not



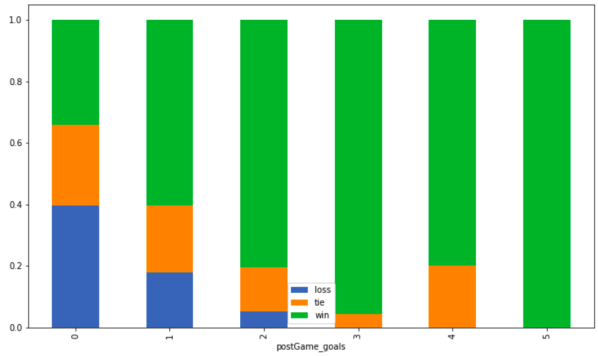
(b) weather a player is playing for the home team or not



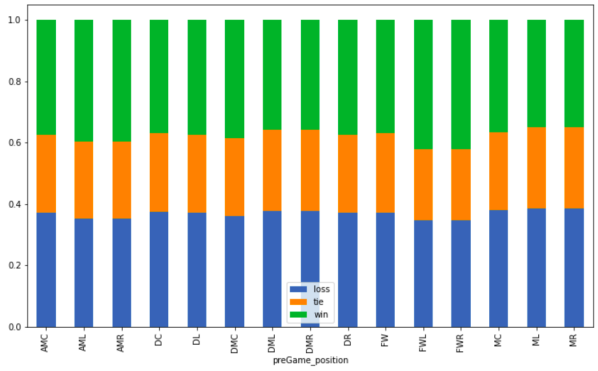
(c) pre-game potential from the video game features



(d) the number of performed shots by a player

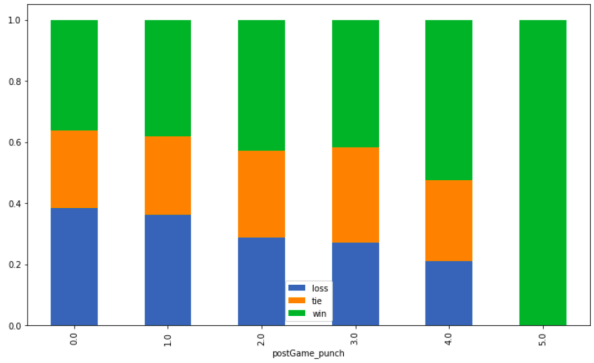


(e) the number of scored goals by a player

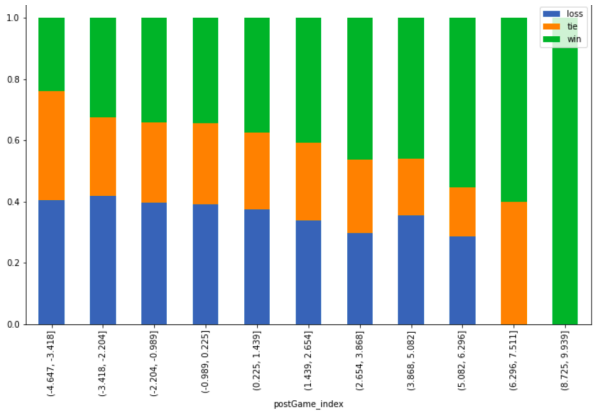


(f) the position which a player plays in

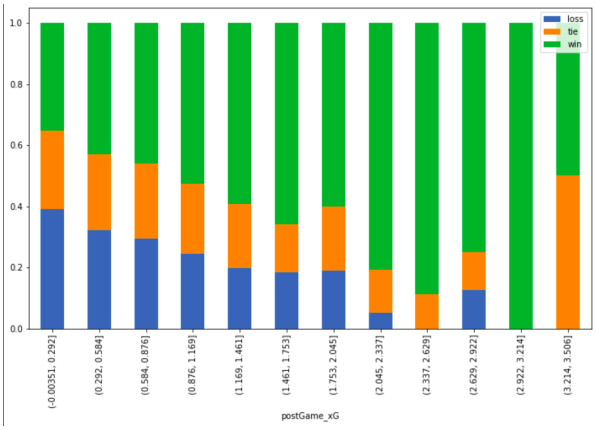
Field players who received a red card were half likely to win(figure 7a). Home players are almost 50% more likely to win(figure 7b). Players with video game rating of more than 73 seem to be more likely to be on the winning side(figure 7c). The more shots(either kind) a player has had, the more likely his team is to win the match(figure 7d). Players scoring 3 or more goals have never been on the losing side(figure 7e). Players with higher performance indices(player ratings including xG and Stephenson and pre-game expected



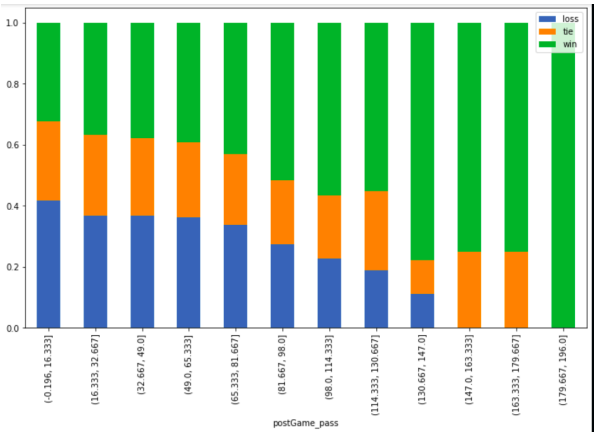
(g) the number of punches by the goalkeeper



(h) the stephenson index of a player



(i) the xG index of a player



(j) the number of passes given by a player

Figure 7: Some properties of the players in the data and their relation to the target variable

xG) have higher chances of winning(figures 7h, 7i). Players with more passes(either kind) also have higher probabilities for winning(7j). Goalkeepers having more punches are more likely on the side of the winning team(7g). Other attributes such as position(figure 7f) do not reveal any information about the result.

3.2 Multivariate Relations

Pearson correlation heatmap between the attributes indicate correlation among different measures of pass and also among goals, index and xG ratings(figure 8, figure 9a). Multivariate analysis further indicate relationship between market value, potential and overall which might incline us to drop either ones since increase in any of these attributes lead to increase in the value of their corresponding attributes as well(figure 9b).

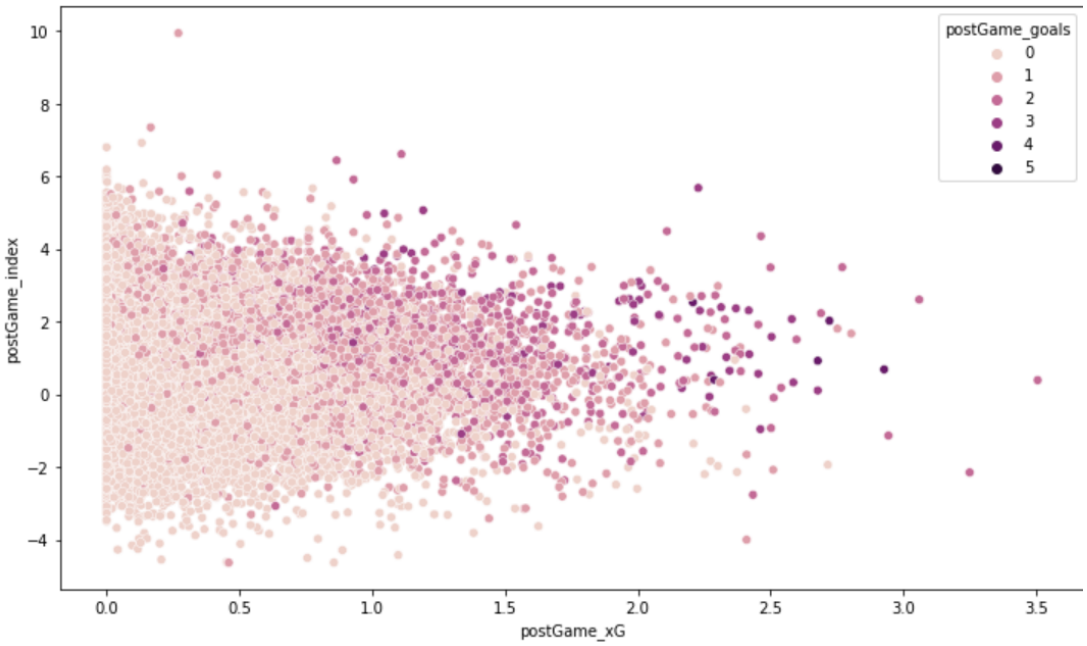
Figure 8: The heatmap of correlations between the features of each player

3.3 Binning Based on Predictive Value

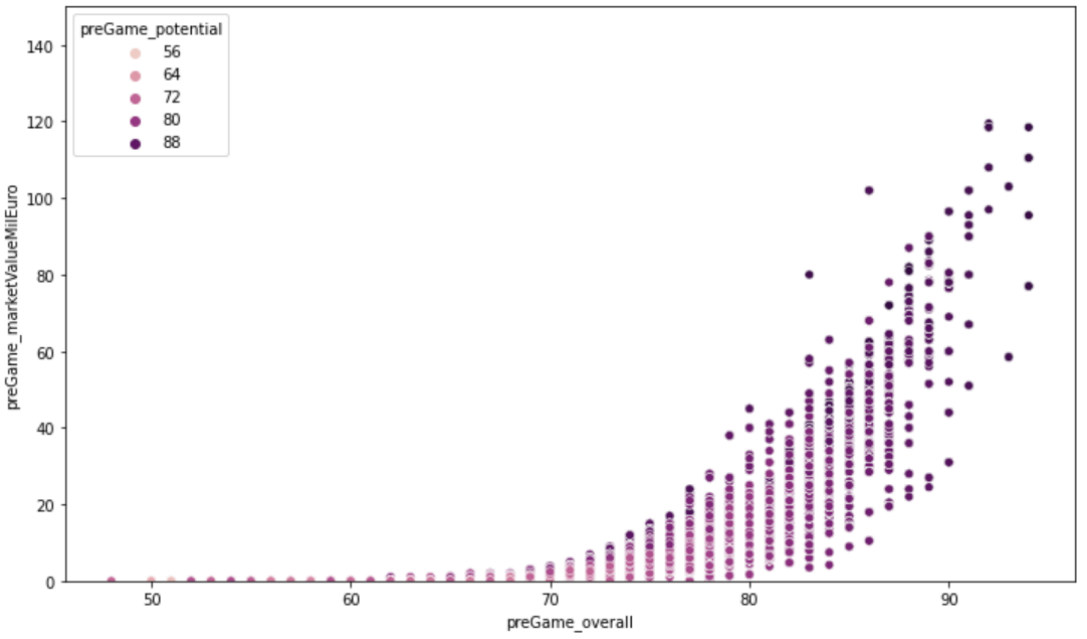
Using the analysis from the predictive variables and the target variable, custom bins were derived with a focus on the predictability of each bin with regard to the target variable. Some of the binned variables and their relation to the target variable is depicted in figure 10.

3.4 Extracting New Features

As explained earlier in the problem understanding phase, the post-game attributes are not available prior to a match taking place. However, by using a window of the values of these attributes across previous matches, a rich attribute could be derived. Each player's performance in the last 4 matches has been averaged over to provide these attributes. some of the variables created are shown in figure 11.

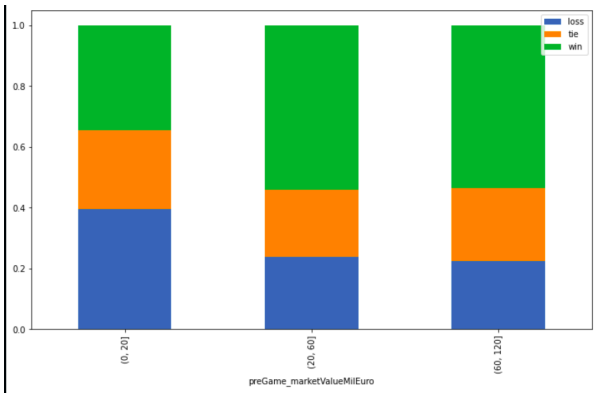


(a) the relation between stephenson index, xG index and the number of goals of a player

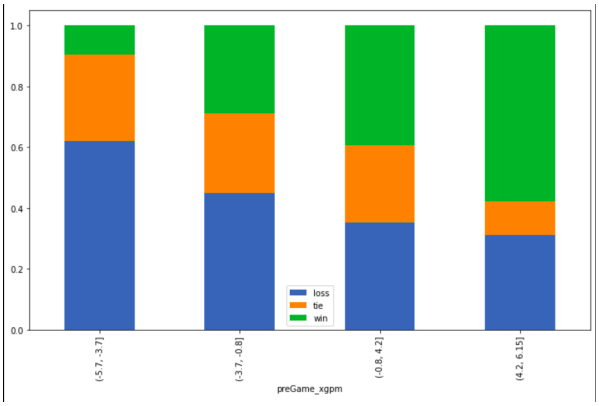


(b) the relation between overall, market value and potential video game scores

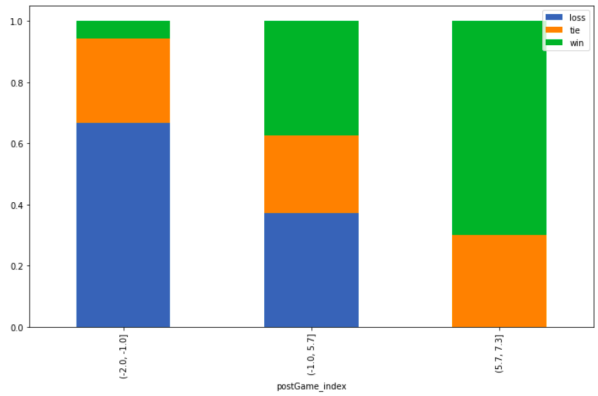
Figure 9: Multivariate correlations between some columns of player data



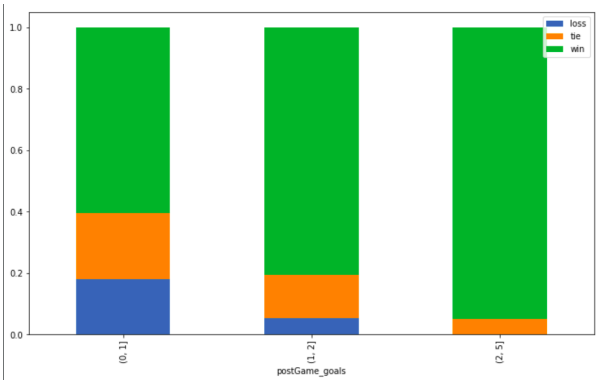
(a) Players' market value



(b) players' xGpm index



(c) the Stephenson index of the players



(d) the number of scored goals by a player

Figure 10: Some properties of the players after binning based on the predictive value and their relation to the target value

	agg_postGame_error	agg_postGame_clearance	agg_postGame_index	agg_postGame_shots	agg_postGame_shots_on_target	agg_postGame_shots_left_foot
80075	0.0	2.25	1.167869	0.5	0.25	0.0
80076	0.0	1.25	1.737601	1.5	0.75	0.5
80077	0.0	1.75	0.717848	1.0	0.75	0.5
80078	0.0	6.75	0.391377	0.0	0.00	0.0
80079	0.0	0.00	1.234975	0.0	0.00	0.0

Figure 11: Some of the aggregated post-game features that are now available before each match of a player

4 Setup

4.1 Cross-Validation

For the cross-validation procedure, the stratified k-fold cross-validation method is chosen. Because the dataset is relatively small comparing to the complexity of the task at hand, we preferred not to perform leave-out validation; instead the k-fold process gives us the opportunity to train and test the model on the entire dataset. With the stratification when splitting the dataset into different folds, the distribution of the class labels in the data is kept throughout the training and validation. The K parameter is chosen to be 5.

4.2 Class Balancing

Class labels are distributed roughly in a 0.45, 0.25 and 0.3 for the home team winning, drawing and losing respectively. One idea to account for this imbalance in the target variable is to artificially up-balance the data records whose target label is of the less present classes. Since our modeling is of loss-optimizing type, one simple approach to have higher than one coefficients for the loss of data records of the less present labels. For instance, the weight for misclassifying a match that resulted in the home team losing could be scaled up to 1.5; Doing so will have the effect of having the home team losing and winning class labels have the same balance. Empirical results however show the opposite, i.e. not balancing the target variable had better results and we chose not to balance the dataset.

5 Baselines

The baseline model of this task is the "null model". Basically for each match, predicting the home team winning is on average 0.45 accurate.

Bookmaker agencies provide betting odds prior to a match. They assign real numbers to each possible outcome and if an individual predicts that outcome correctly, the agency pays him/her the bet money at the magnitude of that real number odd. In order for theses agencies to maximize their profit, they try to assign the lowest odds to those outcomes deemed more probable. One prediction model is to predict the outcome of each match as that outcome with the least odd.

6 Modeling

6.1 Chosen Algorithms

6.1.1 Team Blade-Chest

First, a feature vector for each team is extracted from the dataset. Then the feature vectors of home and away teams h and a, are used as input feature vector to the Blade-Chest model: A linear transformation is used to map the embedded feature vector of

teams to the space of blade/chest vectors:

$$\begin{aligned}
\mathbf{h}_{\text{blade}}\ (\mathbf{x}_h) &= f\left(B\mathbf{x}_h\right) \\
\mathbf{a}_{\text{blade}}\ (\mathbf{x}_a) &= f\left(B\mathbf{x}_a\right) \\
\mathbf{h}_{\text{chest}}\ (\mathbf{x}_h) &= f\left(C\mathbf{x}_h\right) \\
\mathbf{a}_{\text{chest}}\ (\mathbf{x}_a) &= f\left(C\mathbf{x}_a\right)
\end{aligned} \tag{7}$$

where f is the element-wise tanh() activation function, B and C are learnable matrices used to transform team feature vectors into blade or chest vectors respectively. The blade and chest vectors are then used to calculate the matchup score as follows:

$$m_{h,a} = \mathbf{h}_{\text{blade}} \cdot \mathbf{a}_{\text{chest}} - \mathbf{a}_{\text{blade}} \cdot \mathbf{h}_{\text{chest}} \tag{8}$$

where "." is the inner product operation. We then use a linear model with a Softmax activation function to transform the matchup score which is a scaler value to 3-dimensional vector:

$$P(y^0_{h,a}, y^1_{h,a}, y^2_{h,a}) = Softmax(Rm_{h,a}) \tag{9}$$

where R is a learnable 3×1 matrix. The output is an estimate of the probability of the 3 possible game results, i.e. the home team wins, draws and the away team wins.

6.1.2 Player Deep-Set Aggregation

A team can be considered as an unordered set of players. The following form of aggregation function is a universal set function approximator:

$$\mathbf{z}_{\mathcal{A}} = \text{MLP}_{\theta}\left(\sum_{a \in \mathcal{A}} \text{MLP}_{\phi}\left(\mathbf{x}_a\right)\right), \tag{10}$$

where MLP_{θ} stands for a multilayer perceptron with an arbitrary depth and some trainable parameters θ, $\mathcal{A}$ is a set, a an element of $\mathcal{A}$ and $\mathbf{x}_a$ is its feature vector, and $\mathbf{z}_{\mathcal{A}}$ is the single embedding of the set. In Deep Set model, using equation 10, a feature vector is obtained for each team, which is the aggregation of its players' feature vectors. The vectors of all 11 team players are aggregated element-wise and fed to the MLP_{θ}, which takes it to another latent space to produce team embedding z. This vector z is calculated for the home and away teams, concatenated together and is used as the input vector of another MLP with arbitrary depth. A key note to take is that the vector of the home team is always on the same place when concatenating the team vectors to maintain the home advantage phenomenon.

6.2 Comparison with Baseline

As mentioned before, the "always home wins" model has 45% accuracy. In the early stages of testing, our models showed promising results of 48% and 49% for the Blade-Chest and Deep-Set Aggregation models respectively. These accuracies improved after hyper-parameter tuning.

6.3 Hyper-Parameter Tuning

The models hyper-parameters are tuned in a grid-search fashion of all the combinations of the parameters below. The parameter that resulted in the best validation accuracy is highlighted with an underline.

- Dropout: [0, **0.25**, 0.5]
- Dense sizes: [[6], **[6, 6]**]
- Epochs: [15, **20**]
- Learning Rate: [1e-3, **3e-3**]
- Player hidden sizes: [**[8, 8]**, [10, 14]]
- Team hidden sizes: [**[8, 10]**, [14, 14]]

7 Evaluation

7.1 Evaluation Metrics

7.1.1 Accuracy

The accuracy of predicting the outcome of matches in the test set is evaluated for different models. the results are shown in table 1

Table 1: The Accuracy of baseline and proposed models

Model	Accuracy
Always Home Win	45.37%
Bookmakers Odds	48.60%
Team Blade-Chest	49.30%
Player Deep-Set Aggregation	50.45%

7.1.2 Ranked Probability Score

If the observed result of a football match is a home win, predicting a draw is more accurate than predicting an away win. In other words, the output variable in football prediction is ordinal, not nominal. Ranked Probability Score (RPS), a scoring system for predicting ordinal variables, is often used to evaluate football predictions because it considers the order of results. It has been shown that RPS performs better than other criteria such as Brier in assessing football predictions. In addition, since true odds minimize the RPS' expected value, the RPS is a highly reliable scoring system.
Suppose r is the number of possible outcomes (for example in football r = 3, win, draw and loss) and $\mathbf{p} = (p_1, p_2, p_3)$ is the predicted probability vector, for example p_1 is the probability of winning, p_2 is the draw probability and p_3 is the probability of losing. It is clear that $p_j \in [0, 1]$ for $j = 1, 2, 3$ and $p_1 + p_2 + p_3 = 1$. Let $\mathbf{y} = (y_1, y_2, y_3)$ denote the

vector of real, observed outcomes for win, draw and loss. Similarly, $y_1 + y_2 + y_3 = 1$ The RPS for a prediction is defined as:

$$RPS = \frac{1}{r-1} \sum_{i=1}^{r-1} \left(\sum_{j=1}^{i} (p_j - y_j) \right)^2 \tag{11}$$

As can be seen, RPS represents the difference between the cumulative distributions of the model predictions and the actual observations. A prediction's RPS value is always in the range of [0, 1], the smaller the value, the better the prediction. Average over all RPS for all matches in the test set is defined as:

$$\text{RPS}_{\text{avg}} = \frac{1}{|T|} \sum_{i=1}^{|T|} \text{RPS}_i \tag{12}$$

where $|T|$ is the total number of games in the testing set.

The Ranked Probability Score of baseline and proposed models is shown in table

Table 2: The RPS of baseline and proposed models

Model	RPS
Always Home Win	0.420
Bookmakers Odds	0.331
Team Blade-Chest	0.280
Player Deep-Set Aggregation	0.212

7.2 Error Costs

The nature of football match outcome prediction is non-critical per se. Due to its high unpredictability, people find it exciting and an amusement. The risks of a model misclassifying an upcoming match are relatively low if the stakes are low. Betting on football matches could be a disorder and result in heavy losses of fortunes. Relying on a model to bet money on a particular outcome is a high-risk behavior and not advised as no model can be accurate enough to have an expected profit and almost always a model will result in a loss.

7.3 Returning to Previous Phases

Both the Blade Chest model and the Sett Aggregator model have built-in batch normalization, meaning that each vector of data, no matter the space, is standardized.

We had only one categorical attribute, the position, which we omitted for the Blade Chest model since this model does not use the lineup information and also for the Set Agg model since this model views each team as an unordered list.

Binning numerical attributes seemed not worth it; due to our models being linear, we would have to convert these new categorical bins to real values and therefor we decided not to bin numerical variables.

For the Set Agg model, we decided to capture time evolution through using the performance of each player in the last 4 matches. For each player's first match in the dataset, since we have no prior matches, the value becomes null. We decided to make these values zero because of the added effect of regularization that it has.

7.4 Best Model and Hyper-Parameters

The hyper-parameters resulting in the most accurate model is shown in the hyper-parameter tuning section. Also, the metrics reported in this report are for the model that has been trained using the aforementioned hyper-parameters.

7.5 Discussion and Future Work

The Blade Chest model tries to capture the interactions between teams through two feature vectors each encompassing the offensive and the defensive strength of teams which is similar to the real world analysis. The biggest drawback for this model is its ignorance towards players and team lineup.

The Set Agg model's strength lies in its ability to include team players and their previous performances in both spatial and temporal contexts. However, the temporal aspect of modeling could also be modeled through machine learning approaches. One suggestion to improve the Set Agg's ability to capture the evolution of players through time is to use a temporal graph structure where each player has a chained structure spanning through time. The spatial context could easily be modeled with graph structure, forming a spatio-temporal message passing graph.