CS230

Football Match Prediction using Deep Learning (Recurrent Neural Network)

Ahmed Amr Awadallah
Stanford University
ahmedamr@stanford.edu

Raghav Khandelwal
Stanford University
raghav68@stanford.edu

1 Motivation

Football being one of the world's most popular games has a craze in everyone's mind. A football fan must have wanted to know the results before the game at least once in their lifetime! In sport prediction, large numbers of features can be collected including the historical performance of the teams, results of matches, and data on players, to help different stakeholders understand the odds of winning or losing forthcoming matches. The aim of this research is predicting football matches using deep learning algorithms such as DNN and RNN.

2 Problem Statement

Competitive sport is a number game, and modern athletic teams are truly beginning to embrace this fact. We are seeing a surge of data driven jobs in the back office of myriad different franchises, and people are beginning to wonder just how effective cutting edge AI, ML and DL techniques can be when applied to athletic contexts. Sports have always been a purely human endeavour and there is an unarguable element of randomness and chance that dictates who the victor will be on any given day but this begs on an interesting question: is there an underlying pattern to this randomness?

In this project we focused especially on matches in Premier League (England Football league). Few reasons made us work on that topic. We are football fans and also passionate about the area of statistical modelling and prediction. Working on this topic was a good opportunity to join these two interest areas.

Here, what exactly did we predict ? For each game, we will predict 3 probabilities : probability that the home team wins, probability that the result is a draw and probability that the home team loses the match. These predicted probabilities could be useful in evaluating our models' accuracies by forecasting the result of each match, which will be associated with the biggest predicted probability. We used logistic regression, random forest classifier, a 2-layer neural network and an LSTM to output a predicted result which could take any of the three options : a home win, a draw, or win, or an away win

3 Dataset and Features

We have analysed data from different sources corresponding to Premier League [3] matches results from 2008/2009 season to 2018/2019 season. As mentioned we will be working with data of last 10 seasons of Premier league which means we are currently looking into 4180 rows x 78 columns, 36 teams which participated in last 10 seasons and taking 40 features into account from the dataset (Date, HomeTeam, AwayTeam, FTHG, FTAG, FTR, HTAG, B365A, B365D, B365H, BSA, BSD, BSH, BWA, BWD, BWH, GBA, GBD, GBH, IWD, IWD, IWH, LBA, LBD, LBH, PSA, PSD, PSH, SBA,

,

SBD, SBH, SJA, SJD, SJH, VCA, VCD, VCH, WHA, WHD, WHH) which will be used towards manipulation and extracting 38 features (refer appendix 2).

For each match we had access to : the country id, the league id, season, stage, date, match api id, home team id, away team id, home team goal, away team goal, home player X positions, away player X positions, home player Y positions, away player Y positions, home player ids, away player ids, and various betting odds (B365A, B365D, B365H, BSA, BSD, BSH, BWA, BWD, ...)

For each team we had access to : the team id, the team name, FIFA id, FIFA data,and FIFA statistics including build up play speed, build up play dribbling, and more. We decided it to just use it for ELO ranking and correlation as the data per team was inconsistent and had few missing values.

The final features we used 24 features and they are Home team, away team, home team elo score, away team elo score, home team days since last match, away team days since last match, home team win rate, away team win rate, away team draw rate, home team draw rate, Last 7 home team win rate, Last 7 home team lose rate, last 7 home team draw rate, Last 7 away team win rate, Last 7 away team lose rate, last 7 away team draw rate, Last 12 home team win rate, Last 12 home team lose rate, last 12 home team draw rate, Last 12 away team win rate, Last 12 away team lose rate, last 12 away team draw rate, last 5 home team home win and last 5 away team away win . We also augmented this with data on the current season, where for each game, we calculated how many wins, losses and draws the teams had up till that point, as well as their current win or loss streak.

4 Method

A. Logistic Regression : For our baseline algorithm we choose to run a basic logistic regression on our dataset. To do this we initialize a weighted matrix of size (3,24) to avoid issues exploding or vanishing gradient problems, and fit it with our input vector which consists of 24 features (all of home and away matches).

B. Random Forrest Classifier: To capture other than linear and additive patterns in the data gathered, I thought that Random Forrest could be an interesting classifier for this problem. Sckikit-learn implementation of the Random Forrest provides also the predicted probabilities associated with each class

C. Neural Network: The third model we experimented with was a fully connected three layer neural network. Our first hidden layer had 300 hidden nodes and our second had 100 nodes with all of the weights for these layers being initialized to avoid exploding or vanishing gradients. The neural network takes our input vector, and, at each layer, multiplies it with the associated weight matrix for that given layer. Once it has done this multiplication, it runs through the activation function (3) before passing it to the next layer. After passing through the last layer, we use a softmax activation (1) to normalize the output, and then use cross entropy loss (2) to calculate the cost of that iteration. We then used Adam to update our weights to minimize the aforementioned cost. We ran this for 30,000 epochs with a learning rate of 1e-6

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{n} e^{z_k}} (1) \tag{1}$$

$$L(\hat{y}^{(i)}, y^{(i)}) = -\sum_{j=1}^{n} (y_j^{(i)} log(\hat{y}_j^{(i)})) \tag{2}$$

$$R(z) = \max(0, z) \tag{3}$$

D. LSTM : The final model we attempted to use to increase our performance was an LSTM. Our reasoning behind using this model is that football games are sequential in nature but our past models had no meaningful way of capturing this information. With the use of LSTM we hope that this new introduction of information will help increase the performance. The data is manipulated from (Examples, Number of features) to (Examples, Total "Time", Number of Features). In this case we decided to make the total time be 10 matches. This information is then processed for every time slice

(a singular game) for all examples in the dataset. The model itself works as follows: Select time slice, input time slice into LSTM with 16 units, and then input the LSTM output into a softmax densor that outputs 3 units. This output is then evaluated just like all the other models based on a categorical cross entropy loss function.

5 Experiments/Results/Discussion

A. Logistic Regression : As mentioned previously we ran our logistic regression on an input of 24 features, consisting of matches played at home and away. This led to a training accuracy of around 60.95% and a test accuracy of 53.68% which is better than random given our three output classes. It did manage a good spread of results, as can be seen in the below snapshot, but clearly there is room of improvement

```
Logestic Regression
Train Score:  0.6095693779904306
Test Score:  0.5368421052631579
              precision    recall  f1-score   support

           A      0.524     0.526     0.525       329
           D      0.315     0.111     0.164       253
           H      0.575     0.778     0.661       463

    accuracy                          0.537      1045
   macro avg      0.471     0.471     0.450      1045
weighted avg      0.496     0.537     0.498      1045
```

B. Random Forrest Classifier: Classifier's behavior is vastly different as is shown below. The model trained well (78.5%) on the training set but was poor on the test set (40.66%) indicating a variance problem with the dataset. The forest classifier was also able to make predictions on the draw results which logistic regression was unable to do.

```
Forest Classifier
Train Score:  0.7853269537480064
Test Score:  0.40669856459330145
              precision    recall  f1-score   support

           A      0.427     0.647     0.514       329
           D      0.234     0.265     0.249       253
           H      0.558     0.313     0.401       463

    accuracy                          0.407      1045
   macro avg      0.406     0.408     0.388      1045
weighted avg      0.438     0.407     0.400      1045
```

C. Neural Network: To find the optimal neural network we tested a number of alternative architectures, though we kept the depth of the network constant. We used learning rates of 1e-6. We considered 3 different options of the node (20 ,10), (100, 200) and (300,100), and we ran all of these networks for thousands of iterations to determine which gave us the best result. We compared them based on their accuracy on the development set, and our best performer had a learning rate of 1e-6, a node structure of (300, 100) and was run for 30,000 iterations. This initial implementation gave us training accuracy of 61.46% and a test accuracy of 56.36% as shown in the graph below

```
Train accuracy: 0.614673 Val accuracy: 0.56363636
              precision    recall  f1-score   support

           A      0.525     0.556     0.540       297
           D      0.264     0.158     0.197       247
           H      0.660     0.768     0.710       501

    accuracy                          0.564      1045
   macro avg      0.483     0.494     0.483      1045
weighted avg      0.528     0.564     0.541      1045
```

Despite this interesting development, we believe that this demonstrated that past data did have an effect on the accuracy of our prediction, which prompted us to explore how effective an LSTM could be in solving this problem, as it takes into account previous information by default when making its prediction.

D. LSTM: The LSTM overall performed the best out of all our models. It's clear this model still has a bias problem like the rest of the models. The features acted upon are simply not enough to make accurate predictions. It's also interesting to note that the draw performance of this model is pretty weak compared to all the other models.

```
Train Score:  0.5947368421052631
Test Score:  0.5824561403508772
              precision    recall  f1-score   support

           A      0.563     0.617     0.589       345
           D      0.192     0.040     0.066       253
           H      0.621     0.814     0.704       542

    accuracy                          0.582      1140
   macro avg      0.459     0.490     0.453      1140
weighted avg      0.509     0.582     0.528      1140
```

6 Conclusions/Future Work

Model	Train Accuracy	Test Accuracy
Logistic Regression	0.6095693779904306	0.5368421052631579
Forrest Classifier	0.7853269537480064	0.40669856459330145
Neural Network	0.614673	0.56363636
LSTM	0.594736	0.582456

To conclude LSTM was the most successful algorithm that we tested. If we had more time we would try and gather more data on which to train across different leagues, in order to reduce our algorithms' tendency to overfit the training data. Though we had some success reducing this using dropout, an increased amount of data would have been even more beneficial. Additionally, we would like to experiment with different architectures and different LSTM variations, to see if these would improve on our classification accuracy, and maybe allow us to predict not only the result of various games, but also the scoreline of these matches.

7 Contribution

We felt the work was evenly distributed between the team and everyone pulled their weight. Raghav was primarily responsible for doing background research, data extraction and cleaning, feature extraction, and performing various implementations on the algorithms. Ahmed implemented initial variations on the algorithm and actually got the model up and running. We evenly contributed to this final write up. We really enjoyed working together as a team and are proud of what we have achieved during this course.

Appendix 1 Previous Approach : Elo Score

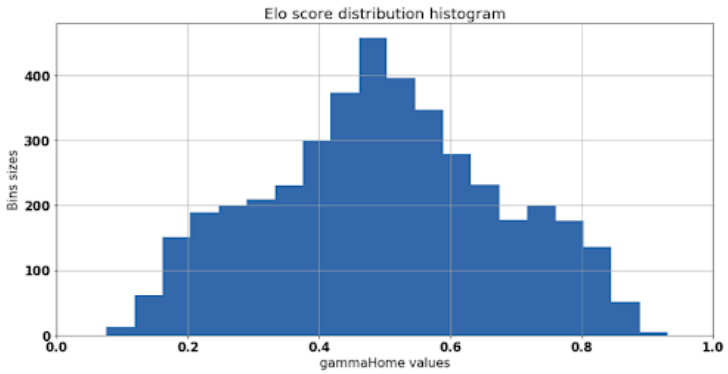

Fig 1 Elo Score distribution histogram

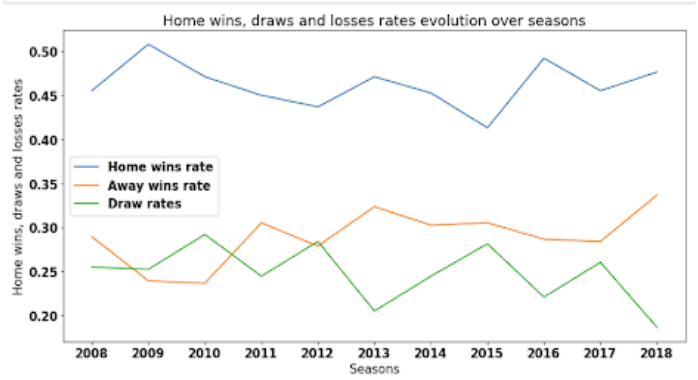

Fig 2 Home wins, draws and losses rates evolution over seasons

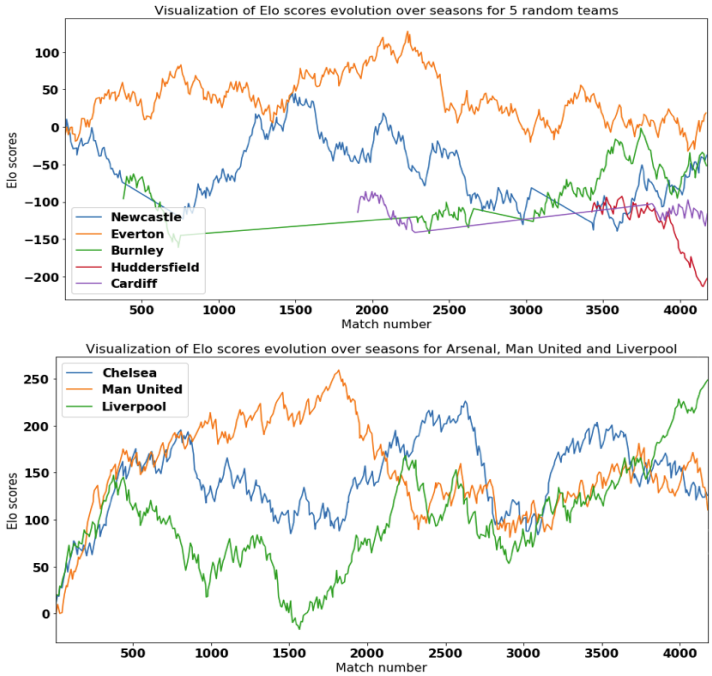

Fig 3: (a) Visualisation of Elo scores evolution over seasons for 5 random teams (b) Visualisation of Elo score evolution over seasons for Arsenal, ManU and Liverpool

Appendix 2 :Features modelling and capturing past performances

Features	Description
1_Last_HTW	1 if home team wins last match else 0, computed thanks to Pandas Series.diff routine
2_Last_HTW	1 if home team wins 2nd last match else 0
3_Last_HTW	1 if home team wins 3rd last match else 0
1_Last_HTD	1 if home team draws last match else 0, computed thanks to Pandas Series.diff routine
2_Last_HTD	1 if home team draws 2nd last match else 0, computed thanks to Pandas Series.diff routine
3_Last_HTD	1 if home team draws 3rd last match else 0, computed thanks to Pandas Series.diff routine
1_Last_HTWH	1 if home team wins last home match
2_Last_HTWH	1 if home team wins 2nd last home match
1_Last_ATW	1 if away team wins last match, else 0
2_Last_ATW	1 if away team wins 2nd last match, else 0
3_Last_ATW	1 if away team wins 3rd last match, else 0
1_Last_ATD	1 if away team draws last match, else 0
2_Last_ATD	1 if away team draws 2nd last match, else 0
3_Last_ATD	1 if away team draws 3rd last match, else 0
1_Last_ATWA	1 if away team wins last away match
2_Last_ATWA	1 if away team wins 2nd last away match
7_HTW_rate	7 last match HT Wins rate, computed thanks to Pandas Series.diff routine on season specific series
12_HTW_rate	12 last match HT Wins rate
7_HTD_rate	7 last match HT draws rate
12_HTD_rate	12 last match HT Draws rate
7_ATW_rate	7 last match AT Wins rate, computed thanks to Pandas Series.diff routine on season specific series
12_ATW_rate	12 last match AT Wins rate
7_ATD_rate	7 last match AT Draws rate
12_ATD_rate	12 last match AT Draws rate
5_HTHW_rate	5 last matches HT Wins at home rate
5_ATAW_rate	5 last matches AT wins away rate

HTW_rate	Home team wins rate from the beginning of the current season
HTD_rate	Home team draw rate
HTL_rate	Home team loose rate
ATW_rate	Away team wins rate from the beginning of the season
ATD_rate	Away team draw rate from the beginning of the season
ATL_rate	Away team loose rate from the beginning of the season
HTHW_rate	Home team "wins at home" rate from the beginning of the season
HTHD_rate	Home team "draws at home" rate from the beginning of the season
HTHL_rate	Home team "lose at home" rate from the beginning of the season
ATAW_rate	Away team "wins away" rate from the beginning of the season
ATAD_rate	Away team "draws away" rate from the beginning of the season
ATAL_rate	Away team "loose away" rate from the beginning of the season

Appendix 3 Correlation graph

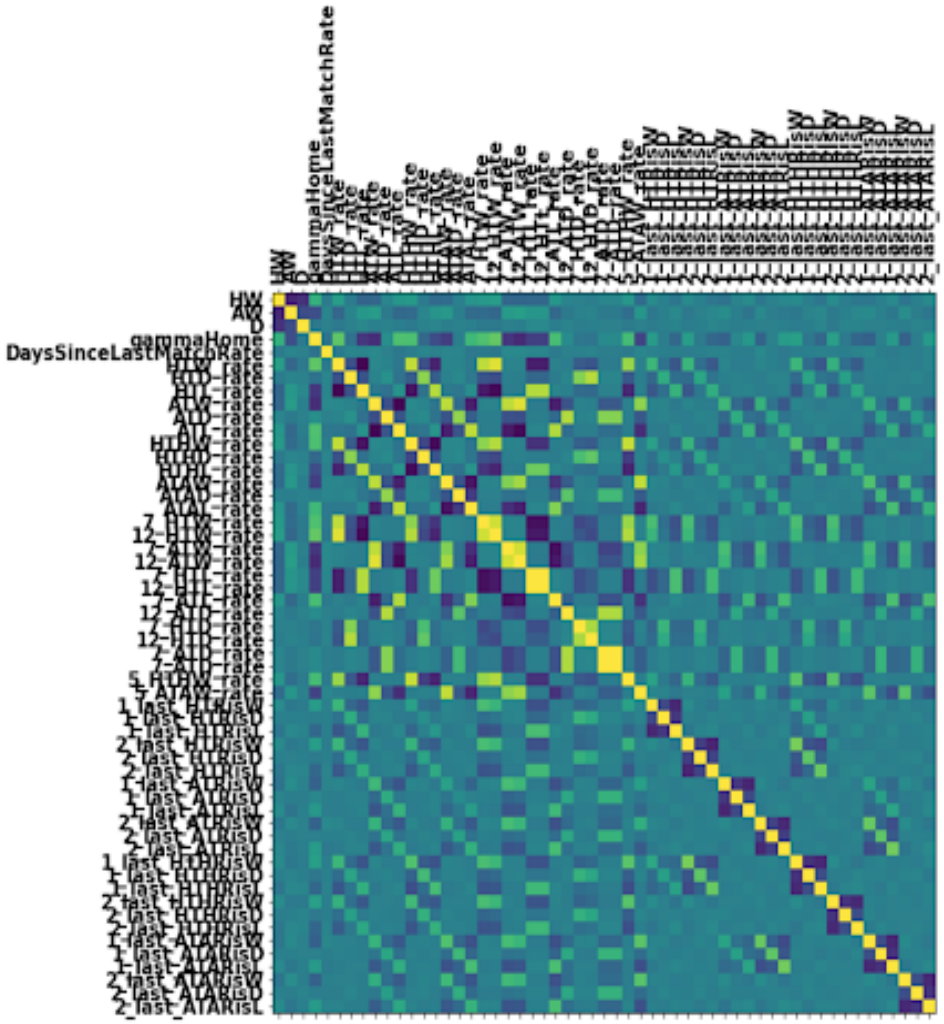

Fig 4: Only a few predictors (gammaHome and long term wins/losses rates) seem to carry relevant information for predicting the results of England League football matches. Moreover, all predictors show very important correlation between each other, hence carry all more or less the same information. This can be a problem when fitting models because this tends to make the training procedure unstable.

References

[1] (2020). Retrieved 9 May 2020, from https://github.com/raghav68/CS230DLProject.git

[2] Mason, S. J., J. S. Galpin, L. Goddard, N. E. Graham, and B. Rajartnam. 2007. Conditional exceedence probabilities. Mon. Wea. Rev., Mon. Wea. Rev., 135:363-372.

[3] England Football Results Betting Odds | Premiership Results Betting Odds. (2020). Retrieved 9 May 2020, from http://www.football-data.co.uk/englandm.php

[4] jalapic/engsoccerdata. (2020). Retrieved 9 May 2020, from https://github.com/jalapic/engsoccerdata

[5] (2020). Retrieved 9 May 2020, from http://www.oddsportal.com/soccer/england/capital-one-cup-2013-2014/results/

[6] football.db. (2020). Retrieved 9 May 2020, from https://github.com/openfootball/

[7] European Soccer Database.(2020). Retrieved 9 May 2020, from https://www.kaggle.com/hugomathien/soccer