GitHub URL

https://github.com/ktadgh/UCDPA_tadghkelly

Abstract

The aim of this project was to use multi logistic regression to predict the result of chess games, based on information available from the PGNs (Portable Game Notation). I would expect players' ELO ratings to be the most valuable predictor, but we also have information on the event type and the round included in the PGN which could also be predictive of the result. This project is based on a trading use-case, so the output I care about is the probability, which could be used to price games or matches and tournaments, using monte carlo simulation.

Introduction

I chose this project use-case as I work for a bookmaking company so sports modelling is relevant to my work. Chess is also a sport I'm interested in and one with a large amount of data, and an official ELO system which is highly valued by players which makes it uniquely well suited to modelling. The ELO system gives an inherent implied value for expected points, where a win is one point and a draw is half a point. However, I want to predict wins and draws individually so that both outcomes can be priced. I aim to predict the probability of a win, draw and loss in a classical chess game, given both players' ELO ratings, the tournament type, and the round.

Dataset

As mentioned above, chess games are generally stored as PGNs so I wasn't able to find a relational database with the information I needed. I tried a number of sources, but most didn't include the time control of the game in the PGN. Chess games can be split into four time formats: Blitz, Rapid, Classical and Correspondence. It's reasonable to expect that results behave very differently for different time controls, with Classical games having more predictable results and more draws generally speaking. I wanted to focus on making predictions for Classical games, so I needed a chess database which included the time control of the game. Yottabase[1] was chosen as it's a large free database with all the information I required. The only issue was that games were only available to download by player, so I downloaded games for the current top 20 by classical live rating[2], converted each to a pandas dataframe with the relevant information, and merged these into one database.

Implementation Process

Database generation

I used the python package chess.pgn[3] to parse the pgn files. This represents each games as a tree of moves. For the purposes of this project the moves are not important, so I wrote a function which iterated through the pgn games, appended the value I needed to a list for each category, created a dictionary with the lists and a header for each list, and converted this to a dataframe and saved that dataframe to a csv. I used lists to preserve the order to ensure that the values matched up across the columns. I used a while loop so that the function wouldn't depend on the number of games in the pgn, which I would not have knowledge of before parsing. I then ran the function on all 20 of the PGNs.

---

[1] https://www.yottachess.com
[2] https://2700chess.com
[3] https://python-chess.readthedocs.io/en/v0.2.0/pgn.html

Data Preparation

In the data preparation section I merged the csvs together to make a database of games, using pandas concat. I removed all non-classical games from my database. I also removed team games, since I am interested in individual events and I believe that they could skew data due to slightly different incentives for players. I did this using a RegEx search. I also removed any null values. I replaced the standard strings used in PGNs to signify results with the points associated with the result for the player with the white pieces, using RegEx. Lastly I added a column to the database with white's ELO-implied expected points.This was calculated according to the formula included in the reference below.[4]

Exploratory Data Analysis

The first thing I wanted to look at was the relationship between the difference between the players' ELO ratings and the points won by the player with the White pieces. I visualised this using a violin plot to get an idea of the distribution for each result (loss, draw and win). It was clear to see that the ELO Difference (White ELO - Black ELO) increased with white's points. Next, I cut the data into bins, so that I could plot the *average* observed points on the x-axis. This showed a clear positive correlation between points and ELO difference (White ELO - Black ELO), but the relationship didn't look linear. Lastly, I did the same but with bins of ELO-implied expected points on the x-axis. This looked more like a clear linear relationship. I calculated the mean squared error and mean absolute error of the ELO-implied expected points versus the actual points, to use as a benchmark to compare my model with.

Multinomial Logistic Regression

To make predictions in the three categories of Win, Draw and Loss, multinomial logistic regression rather than binomial logistic regression was used. I added variables of the squares of the difference, as I thought that the magnitude of the difference might impact the model, regardless of in whose favour. I also added squares of white's ELO-implied expected points and White's ELO, in case those were more closely associated with the result than the original values. The points values were doubled to 0, 1 and 2. The data was split into a train and test set, and a logistic regression model was fit on the training data. The accuracy and log-loss was calculated for the model, accuracy was reasonably high considering there are three possible outcomes, but the log-loss is of more interest.

The event type was added as a categorical variable, and rounds were converted to numeric so that they could be added to the model. It would make intuitive sense that both of these could have an impact on the outcome of the game; if the game was part of a swiss tournament for example, it may be the case that a player is more incentivised to draw than they would be in a knockout tournament. Similarly, in the final rounds of a tournament there would likely be more must-win situations, so I would expect more draws in the earlier stages.

Cross-Validation and Hyper-Parameter Tuning

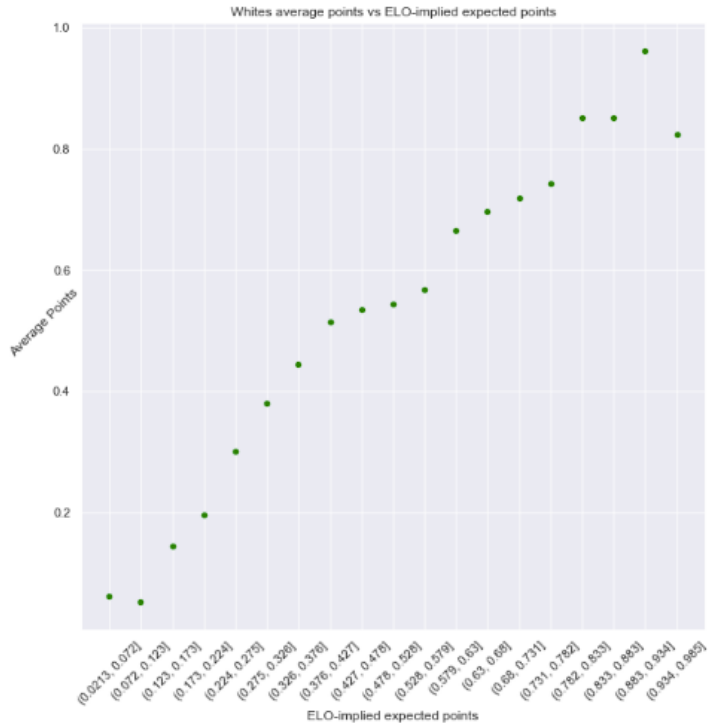
Here round and event type were added to the model, and hyperparameter tuning was conducted using GridSearchCV. Due to the nature of the project, scoring was set to use negative log-loss. First this was run without including the squared values of the ELO difference, the ELO-implied expected

---

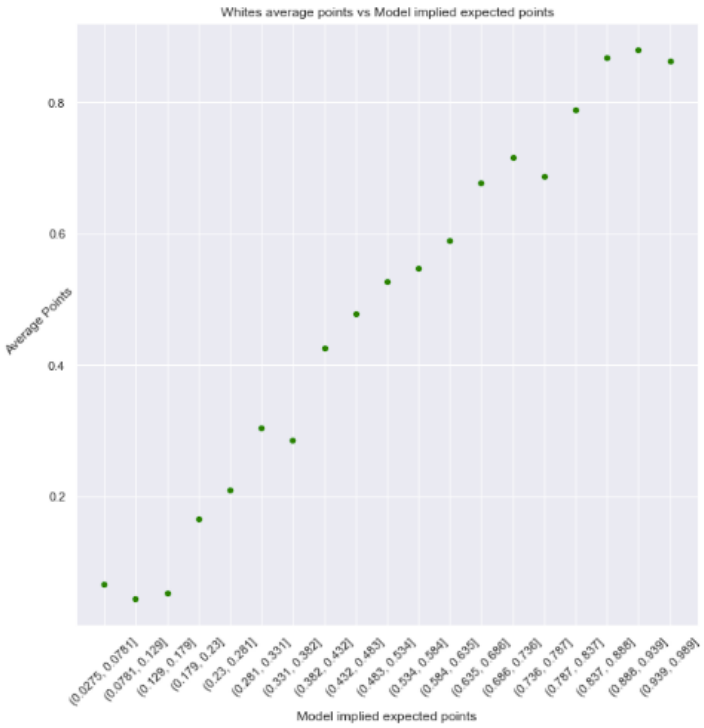
[4] https://www.cantorsparadise.com/the-mathematics-of-elo-ratings-b6bfc9ca1dba

points, and white's ELO. This model improved markedly upon the previous non hyper-parameter tuned model, with a log-loss of 0.891 compared to 0.919. I then ran GridSearchCV again, with the squared variables included. Including the extra information decreased accuracy on the test set, but it outperformed the previous model in terms of log-loss. Since I am concerned with estimating probabilities rather than classifying results, this model was chosen.

Results



ELO-implied expected points (shown above) had a mean squared error of 0.09902 and a mean absolute error of 0.2396 for the test data.



The model's implied expected points had a mean squared error of 0.09608 and a mean absolute error of 0.2351 for the test data, outperforming the ELO-implied expected points.

Insights

1. After examining the first hyper-parameter tuned model, we can see that an increase in rounds is actually associated with an increase in the probability of a draw. This is the opposite of what I expected, but could also make sense due to different dynamics late in tournaments, such as a player only needing a draw to win the tournament.
2. Swiss tournaments overall increase the probability of a decisive result and decrease the probability of a draw.
3. The model associates an increase in white's ELO with a *decrease* in the probability of white winning, and an increase in the probability of a draw. I take this to be a good sign for the model, meaning it is accounting for the fact that higher rated players will tend to draw more since their play is closer to perfect (and chess, with perfect play, is a draw).
4. Some of the coefficients associated with the event types were very strongly associated with the probability of the result, with their coefficients being quite high. The coefficient associated with the elo difference
5. Adding the squared variables improved model accuracy slightly in terms of log-loss, which decreased from 0.891 to 0.887. However, it made interpreting coefficients more difficult.
6. In the new model, the ELO difference has a positive coefficient while its square has a negative coefficient. This would seem to indicate that, for example an ELO difference of -10, implying that Black's ELO is higher than White's, would make a draw less likely than an ELO difference of +100. This is again the opposite of what I would have expected, but it may be explained by the ELO-implied points squared coefficient, which could be counteracting this, as it's ELO-implied points is also a function of the ELO difference.