Using Statistical Methods to Detect Chess Cheaters

Emanuel Lowcock

With the development of technology, detecting chess cheaters by the hardware that they carry has become harder. It is estimated to cost chess tournaments 10x more to be able to detect the hardware cheaters are using than the cost of the cheating hardware itself. Because of this, the development of statistical methods to predict when a player is cheating at chess (i.e. using a chess engine to know what the best move in the position is) is paramount to preserving the integrity of chess competitions.

Predicting if someone is cheating is, at its root, a classification problem. The scope of this project is to develop methods which are able to predict if white (i.e. the player of the white pieces) or black (i.e. the player of the black pieces) is cheating. The ideal scenario would be to have a dataset with games where humans are playing against each other, labelled with if white, black or neither were cheating. Unfortunately, such datasets are not available to the public - chess cheaters are reluctant to identify themselves publicly and chess platforms with sophisticated cheater-detection methods do not share their own datasets to minimise the risk of cheaters learning how they are being detected.

To circumvent this issue, games were collected where humans played against a chess engine and the engines were labelled 'cheaters'. Games were also collected where titled chess players (high-rated players) played against one another and were assumed to be games where neither player was cheating.

The games were collected from the website 'FICS games' (www.ficsgames.org), the free chess database.

For this project, the classification models selected were logistic regression, K-nearest neighbours (for the first two rounds of modelling), decision tree, random forest and XGBoost.

Accuracy was selected as the overall evaluation metric, though future work should investigate prioritising the F1 score, a harmonic mean of the precision and recall scores because a balance must be found between minimising the number of cheaters who are not detected and the number of non-cheaters who are flagged as cheaters.

Cleaning and Preprocessing

The final dataset was created by following several steps to eliminate unneeded columns and engineer new features. Games that were played by computers against humans and humans against humans were combined and then filtered by only keeping games where each player's elo was within the bounded range of 1600<=elo<=2857. This was done in an attempt to negate the difference in elo distributions of both the computers and humans. Two features were engineered from the dataset: the percentage of moves of each player that were the same as an engine's top recommendation (the engine used by the author to evaluate all positions in every game was Stockfish), and the average time per move for each player. The reason for choosing these features is that humans tend to make many small inaccuracies throughout a game and thus will have a lower percentage of moves that match an engine's top recommendation, and humans tend to take longer when calculating what the best move is in difficult position whereas computers will take the same amount of time regardless of the difficulty.

Insights and Conclusions

The first two rounds of modelling ran into problems due to underlying issues with the collected data - primarily that the distribution of elos between humans and computers was so different. By adding games where humans played against other humans and removing each player's elo the accuracy decreased for all models tested.

Table 1 - Results

	LogReg	KNN	DT	RF	XGBoost
CvH	0.909	0.9458	0.9343	0.9703	0.9587
CvH&HvH	0.8701	na	na	na	na
NoElo CvH	0.7532	0.8428	0.8455	0.8927	0.9286
NoElo CvH&HvH	0.6911	0.8126	0.8554	0.896	0.9093
Feat Eng CvH&HvH	0.89	na	0.9336	0.9882	na
NoElo Feat Eng CvH&HvH	na	na	0.8631	0.9541	0.9549
Grid Search Feat Eng CvH&HvH	0.89	na	0.9747	0.9863	0.9881

The engineered features of the percentage of moves each side played that was the engine's top recommendation, and the average time spent per move, improved the accuracy of all models tested, even after only taking games within a certain elo range to address the difference in average elo between humans and engines. This is not surprising as games, where an engine is playing against a human, are likely to have a high percentage of moves matching the engine's top recommendation. What is interesting is that for the logistic regression model using the engineered features, the 15 highest value coefficient for each sub-classification model did not have the percentage of moves that were the engine's top recommendation. In predicting if white or black was cheating, the elo and average time per move were still within the top 7 and 6 respectively, so it is likely that not enough was done to address the difference in average elo of computers and humans, and that because the computers will typically spend the same amount of time per move the average time spent is a strong indicator (though 'smart' cheaters will tamper with this metric by consciously moving slower at different points in the game).

For the decision tree, the player's elo, average time per move and percentage of moves that were the engine's top recommendation all were within the top 10 most important features in predicting if a player was cheating. Interestingly, an opening was not the most important feature - instead, time control took the top spot. This suggests in the dataset, the CvH games had a significantly different distribution of time controls to the HvH games. Indeed, even when the elos were removed, all models still performed well due to the engineered features, but the most important features were still the time controls.

Future work should investigate further the openings that were the strongest indicator of if white or black was cheating, as one can infer that computers tend to favour particular openings over others. This is indeed a known trait of modern engines, as their playstyle does differ from human players: engines have been noted to play in a way as to keep their options open as long as possible, whereas humans will seek an early conflict which forces the position into more set possibilities.