Micah Wenger

Dean McGahan

Kerem Acar

**Abstract**

In this project we used data from Lichess.com to predict the openings of Chess games based on the average numerical rating of the two players. We parsed the data to get the information we needed then used it to build a Machine Learning model called a decision tree. We then were able to use that tree to see different lists of openings moves based on the rating we gave it.

**Introduction**

Lichess.com is a popular online chess site that is built on an open source platform and makes data from millions of games available to the public. Since Chess is a deterministic game we wanted to use the data from past games to make predictions using machine learning. We decided to use the average rating of the two players and the list of moves that were played as our input features, and a predicted list of moves as our output features. Because in Chess each move depends on the last, we decided to use a decision tree as our model so that we could look at consecutive chains of moves played. If we treated each move independently we would end up with a list of moves that contains commonly played individual moves that make no sense together. To prevent the decision tree from becoming unmanageably large we limited the depth to the first 10 moves of the game. This problem could have an impact on training for Chess players.

One similar work can be found here <https://towardsdatascience.com/predicting-professional-players-chess-moves-with-deep-learning-9de6e305109e>. The strengths of this work include combining various machine learning techniques to achieve a result. A weakness of this work is that it does not take into account certain details of the game like whether a piece is captured in the middle of the board or the edge.

Another similar work can be found here [http://cs229.stanford.edu/proj2014](http://cs229.stanford.edu/proj2014/Sameep%20Bagadia,%20Pranav%20Jindal,%20Rohit%20Mundra,%20Analyzing%20Positional%20Play%20in%20Chess%20Using%20Machine%20Learning.pdf). A strength of this work is that it is able to improve on the ability to predict the result of a chess endgame over traditional chess engines. This work might be able to be improved with a larger data set.

Finally, this video is an example of someone who tried to solve chess using a computer program: <https://www.youtube.com/watch?v=MFNv-FJFGTg>. One strength of this project is his optimism to think he could possibly solve chess on his own. The project fails because his general approach was impossible to implement, a human just can’t memorize every move that should be played in any position.

The particular problem we chose may not seem important to solve outside of the world of Chess, but any successful application of Machine Learning can be used to solve bigger problems. As far as impact goes, in the future players and coaches may be able to use results like ours to learn more about player habits and improve further.

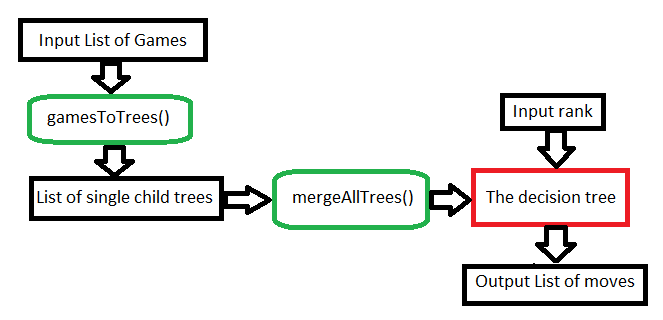
**Our Contribution**

Our contributions include:

* Data Parsing into Objects - Kerem Acar
* Game Objects into single-child trees - Dean McGahan
* Tree merging - Micah Wenger
* Tree file saving - Micah Wenger

**Materials and Methods**

For this implementation of a decision tree, we used python and regex. Regex was used to parse the file line by line in order to create the game objects and we chose python because it makes data handling very easy and is easy to write in. We also used json to export the decision tree in order to save the data because large data sets take a long time to parse.



**Dataset Description**

Our data comes from <https://database.lichess.org/> where there is over 500GB of compressed game data categorized by month. Each data point contains; type of game, website, white/black usernames, the winner, date/time, white/black Elo (rating), the opening, a list of moves, and more. That equates to over 2.5 billion games dating back to the beginning of 2013. So we used 6GB (uncompressed) of game data to train our model.

**Sample Data Point**

| [Event "Rated Classical game"] |
| --- |
| [Site "https://lichess.org/j1dkb5dw"] |
| [White "BFG9k"] |
| [Black "mamalak"] |
| [Result "1-0"] |
| [UTCDate "2012.12.31"] |
| [UTCTime "23:01:03"] |
| [WhiteElo "1639"] |
| [BlackElo "1403"] |
| [WhiteRatingDiff "+5"] |
| [BlackRatingDiff "-8"] |
| [ECO "C00"] |
| [Opening "French Defense: Normal Variation"] |
| [TimeControl "600+8"] |
| [Termination "Normal"] |
| 1. e4 e6 2. d4 b6 3. a3 Bb7 4. Nc3 Nh6 5. Bxh6 gxh6 6. Be2 Qg5 7. Bg4 h5 8. Nf3 Qg6 9. Nh4 Qg5 10. Bxh5 Qxh4 11. Qf3 Kd8 12. Qxf7 Nc6 13. Qe8# 1-0 |

Of the available data points, we only used WhiteElo, BlackElo, and the moves list. White and Black Elo are both quantitative data points because they are scalar values. The moves list is a list of n(changes for each game) elements, where each element is a string, representing a move in chess.

**Algorithms Used/Methodology**

The algorithm we used was a decision tree. Each node on the tree represents a move that a player would make. So the first level is white, then the second is black, then the third level is white again and so on. So the height of the tree is the number of moves that can be predicted by the AI. In each node, we store the ranks of all the moves that have been merged into that node, so we can calculate the average. From there, if we are trying to figure out a moves list based on the rank, we compare the input rank to the average ranks and choose the closest node.

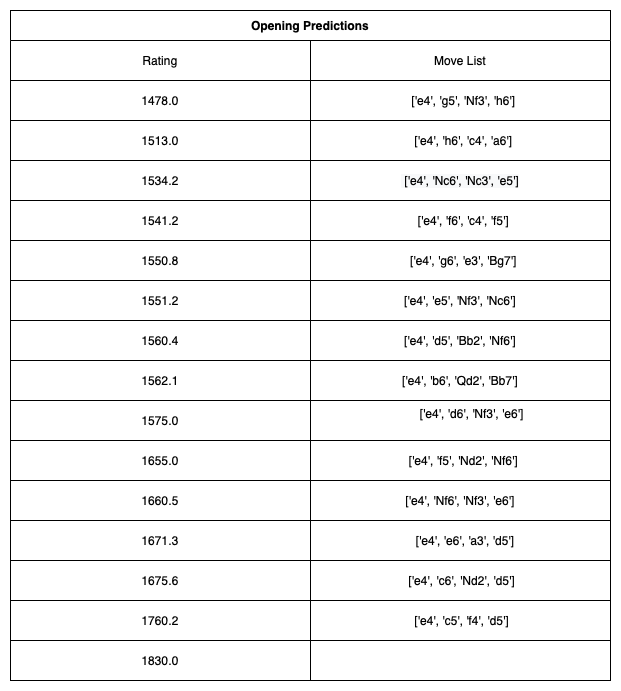
**Performance Evaluation**

We split the data into 80% training and 20% testing. Then we used the model to compare the predicted move list for a rating with the actual move lists played for that rating. We expected a low rate of successful predictions, because our model only predicts the most commonly played opening. For example, if out of 100 games one opening is played 5 times and 95 openings are played one time, the tree will always predict the opening played 5 times but will have a 0.05 success rate.

Our performance evaluation revealed that our test data was too small to show a correlation with training data, so we concluded we need a larger data set in the future.

**Experimental Analysis**

github link: <https://github.com/Uuuuuumm/ChessPredictor>



**Conclusion**

In conclusion, we may need to apply additional machine learning techniques to be able to process a larger amount of data and increase the accuracy of our predictions. However, we have found out that the decision tree model can be used to find which moves are preferred in Chess openings depending on the rating of the players. In the future, we would like to be able to use the decision tree to find the best move possible and use more training data.