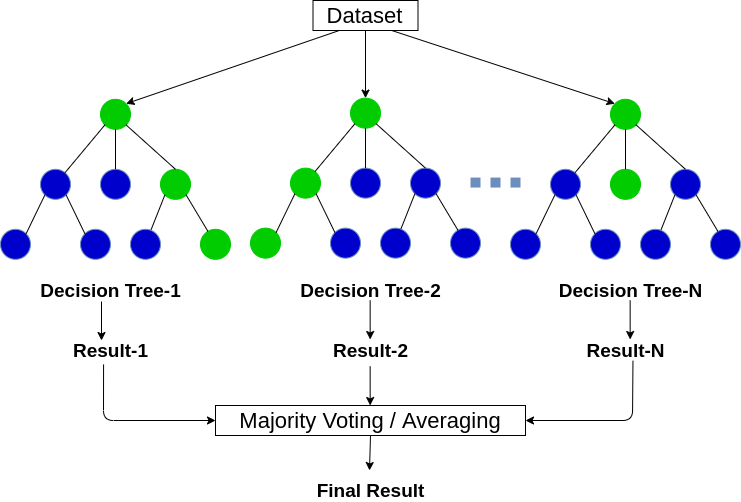
Predicting Chess Matches Based on Opening Moves & Rating

**Progress Report 2**

During the past week, my focus was on implementing the Decision Tree model and training it using my dataset. Additionally, I split the data into an 80/20 ratio for creating training and testing sets.

While implementing Decision Trees, I discovered that they may not be the best model for predicting chess games. Decision Trees can become quite large quickly, as shown in my last progress report. Depending on the project's task and requirements, certain models may be more effective than others. While I am uncertain whether Decision Trees are the best choice for this project, there are other models, such as Logistic Regression, Random Forest, and Gradient Boosting, that could be viable options.

This week, I conducted research on Random Forests, both conceptually and practically in Python. I did this as a contingency plan in case Decision Trees become too complex or do not work well for this project, given the potential for many variables and moves in chess. A Random Forest is essentially a collection of Decision Trees whose results are combined to produce a result. This approach can be beneficial when dealing with complex data that requires the use of multiple Decision Trees.

This is an example of a Random Forest. The individual Decision Trees within the forest compute different results, and the final prediction is obtained by averaging the results of all trees.

To split my dataset into training and testing sets with an 80/20 split, I used the train\_test\_split() function.

This week, I realized that I needed to numerically encode the string of moves in my dataset, which are denoted as strings of Chess Short Algebraic Notation (SAN). To accomplish this, I first considered vectorizing the moves and then putting them into lists. However, I quickly discovered that while SAN appears simple, it is complicated to encode. Each move of a piece is represented by the piece's uppercase letter (except for pawns, which have no letter), followed by the destination square's coordinates. For example, "Be5" represents a bishop moving to the square E5. Additionally, the letter "x" in SAN indicates a capture, so "Bxe5" would mean that a bishop captured a piece on the square E5. While there is much more to SAN, these are the basics.Graphical user interface, text, application

Description automatically generated

I believe that encoding the moves using this approach will help ensure the accuracy of predictions in the future. Although I have not yet normalized the rating data, I may do so later this week to further improve the results.

Moving forward, I plan to try implementing the algorithms this week. If both random forests and decision trees prove to be inefficient or ineffective, I will conduct further research and explore alternative approaches. I'm excited to start working on the implementation of the algorithms this week, as it's an important step in the project. While I'm hoping that the initial implementation will be successful, I understand that it's not always the case. If I run into any issues, I'll take a step back, re-evaluate, and try to identify the source of the problem. I may need to refine my data preprocessing techniques or explore alternative algorithms to achieve the desired results.