

## Contents

Basic Info .....	2
Background and Motivation .....	2
Project Objectives .....	2
Data .....	2
Data Processing.....	3
Visualization Design .....	3
Brainstorm .....	5
Prototype 1 .....	6
Prototype 2 .....	7
Prototype 3 .....	8
Finalization .....	9
Must-Have Features.....	10
Optional Features .....	10
Project Schedule .....	10

## Basic Info

Project title: Easier Openings

Name: Ethan Stanley

Email: u1312964@utah.edu

uID: u1312964

Repository: <https://github.com/ethanstanley3/dataviscourse-pr-mentalhealthfactors>

## Background and Motivation

I chose this project because I enjoy playing chess, but also because I am interested in game strategies that are suboptimal against an optimal opponent, but more effective against imperfect opponents. In particular, many chess players study theoretical openings in order to improve. However, these openings are frequently based off of expert level play. As a result, less experienced chess players may have less success with these openings because they are not forgiving to mistakes. I think it would be cool to make a visualization that allows one to explore which openings are most successful across a range of skill levels.

## Project Objectives

The primary question I am trying to answer is what chess openings are most effective at lower skill levels. I would like a visualization that allows the user to explore what moves are popular or successful from a given position and be able to choose the skill level of the games in the dataset. That way, one could filter for intermediate games to see which openings and moves are effective at the intermediate level, as an example. A benefit is that people, including myself, could study and learn openings that consider the skill level of the players using them.

I am also curious to see if my visualization will reveal openings that are a) only effective at high levels b) only effective at lower levels or c) effective across all skill levels. This would be useful information to know when deciding which openings to study further. The visualization could answer questions such as “which openings will be effective right now?” and “which openings will continue to be effective as I improve at chess?”.

## Data

I will be collecting my data from [https://database.lichess.org/#standard\\_games](https://database.lichess.org/#standard_games).

This is an online database of all chess games played over the internet on lichess.com. There are currently around 3.7 billion games in the database. This dataset is in PGN format, which is a standard plaintext notation for chess games. The dataset is free for anyone to use.

## Data Processing

I expect to do a fair amount of data processing. First, I will have to convert the dataset into a format that is easier to use, such as JSON. I will have to cut out a lot of the extraneous data, such as the usernames of the players. I will also need to decide what data to include as loading 3.7 billion games isn't feasible for my visualization. I plan to derive quantities about what moves are popular from a given position, what moves are statistically the most successful, and filtering the games by skill level dynamically.

To achieve this, I will write a Python script that will convert the dataset into a JSON file with only the relevant data for my visualization. I will then use d3 to group the data based on the having the first n number of moves matching, where n changes as the user interacts with the visualization.

## Visualization Design

I have included 5 sketches on the following pages: a brainstorm of elements to include, 3 prototypes exploring different aspects of the brainstorm, and a finalized version that incorporates the best components of the prototypes.

A general idea I explored in the brainstorm is how the user should interact with the visualization. First, I considered the most obvious mode of a chess board on which the player can move pieces. This would be the easiest input for a chess player to pick up, but it is hard to display meaningful data on the board. It gets cluttered too quickly. I explored this idea in prototype 2. Another option is to represent the possibilities in the chess game as a tree. The user could click on a node and the tree would dynamically expands to show subsequent possibilities. I could encode data (like move popularity or win rates) with the radius of the nodes, width of the edges, or color of the node. But this is problematic because it is hard to look at a path through a tree and connect it to a chess board. I explored this design choice more in prototype 1. Lastly, I considered a nested pie chart, where slices of the pie chart could be clicked and more slices would appear, designating possible moves. The area of the slices would encode the usage or win rate of the move. This idea was explored in prototype 3.

I also want a way for the user to filter the games in the dataset by the skill level of the players. There are four relevant numbers for this: player one minimum rating, player one maximum rating, player two minimum rating, and player two maximum rating. This could be encoded with either a pair of sliders or a resizable and movable brush tool on a chart.

### Prototype 1:

In this prototype, I explored using a tree to represent the moves of the chess game. This has the advantage of effectively showing all past moves, instead of just the current state of the board. The relevant data (move popularity or move win rate depending on what the user is interested in) would be encoded by the widths of the edges or the areas of the nodes. Width is one of the

most effective visual encoders, so it would be effective. Area is not quite as effective, but it could be used to encode the metric that the user is less interested in. For example, area could encode win rate, while the width of the edge could encode usage rate. That way, the marks (nodes and edges) would efficiently encode the relevant channels. One problem with this design is that the screen could become cluttered by having too many nodes. On average, there are 20 possible moves from a given chess position. I included a table at the bottom of the visualization that can include more relevant data about the most recently selected node. This design uses sliders to filter the skill level of the chess games being visualized.

#### Prototype 2:

This prototype explores using a chess board as the main visual component. The possible moves would be encoded by arrows on the board. The relevant variables would be encoded by the width of the arrow. For a less important variable, I could use the opacity of the arrow. Opacity is a much worse channel than area or width, so I think this design less effectively encodes the data. Additionally, it suffers from being cluttered. Many arrows would overlap. This would make it hard to select which move to explore. To overcome the limited effectiveness of the encoding channels, this design would need a table containing relevant data at the bottom.

#### Prototype 3:

I explored a dynamic and layered pie chart in the design. Each slice of the pie chart represents a possible move, where the area of the slice encodes either the win rate or usage rate of the move (depending on the user's preference). When a slice is clicked, more slices appear above the clicked slice that encode possible moves. I have two channels to work with: slice area and slice color. These could be used to encode my two variables. I would use area to encode the more important variable. Although area is not as effective as width for encoding data, the slices of a pie chart will be easier to compare than the width of the edges of the tree found in prototype 1. This is because they share a common axis, unlike the edges of the tree.

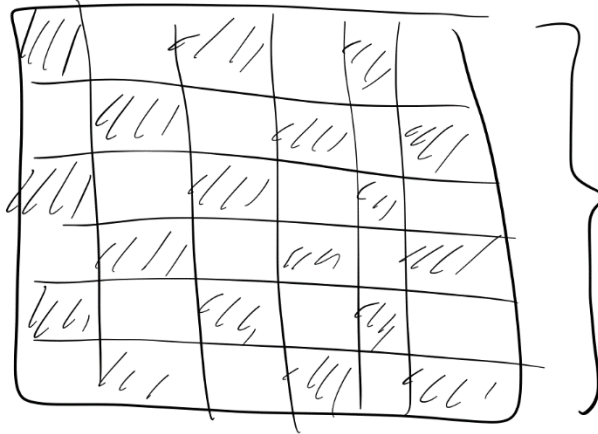
#### Finalization:

From these drafts, I kept the dynamic pie chart because I believe it is the best way to compare all the moves from a given state. I also kept the chess board as a tool for inputting moves and displaying the current state of the board because this is what any online chess player would be most comfortable manipulating. To filter the games in the dataset, the user can drag and resize a rectangular brush too. This is critical to fulfill the visualization's purpose of showing effective openings for any skill level, rather than just experts. I believe this combination most effectively encodes the data about a given move while still being easy to manipulate.

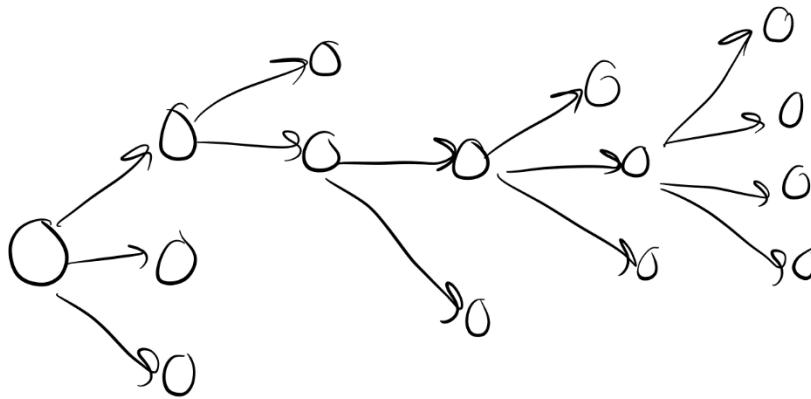
## Brainstorm

Brainstorm:

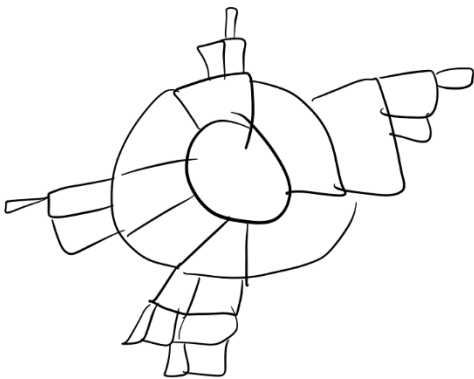
Need a way to let user  
filter data by skill level  
sliders? Dragable / resizable box?



} could have a  
chess board to  
show current  
state and  
easy input



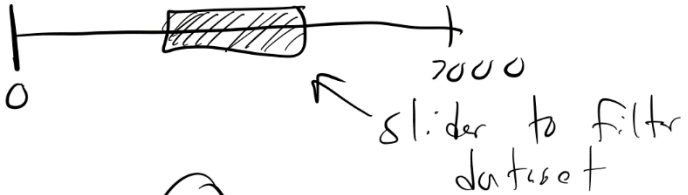
} Interactive tree to show alternatives  
of each board state



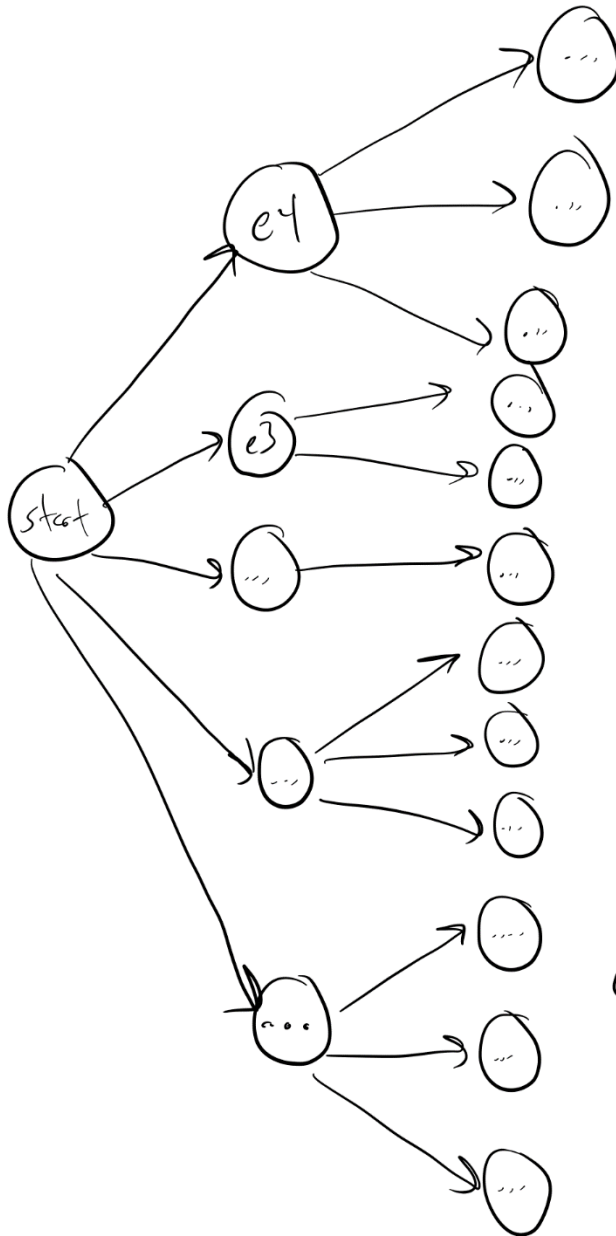
} interactive pie chart  
whose area encodes  
move popularity  
or move win rate

## Prototype 1

Skill range (elo):



0 2000  
slider to filter dataset



Nodes can be clicked on to show their descendants

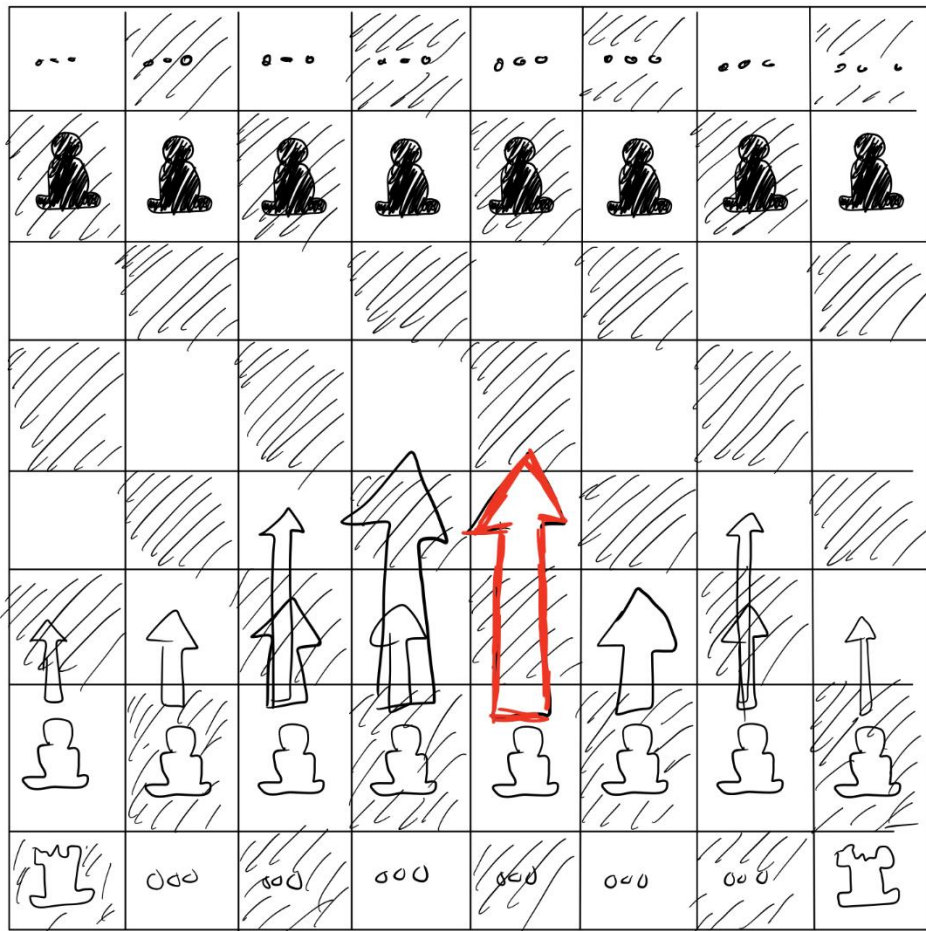
They can also be right-clicked to show statistics about them, such as

- win rate
- play frequency
- usability

Can click on a parent node to go back in the tree

Data about current position also displayed here

## Prototype 2

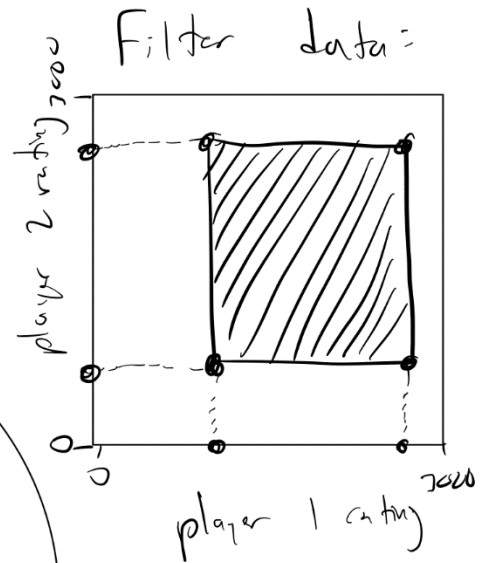
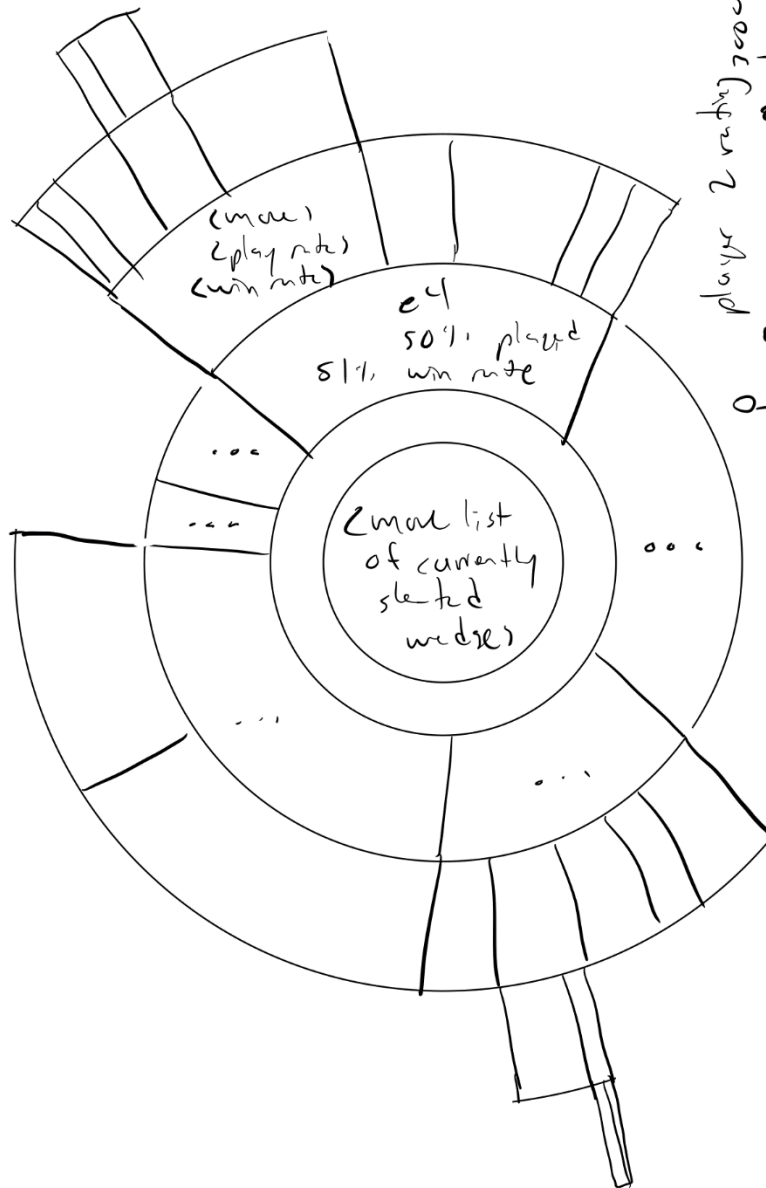


Popular moves

move	f-placed $\Delta$	success rate	avg elo of user	
e4	60	x	x	
e3	10	x		
d4	20	x		
d3	..	x		
...	...	x		
...	..	x		

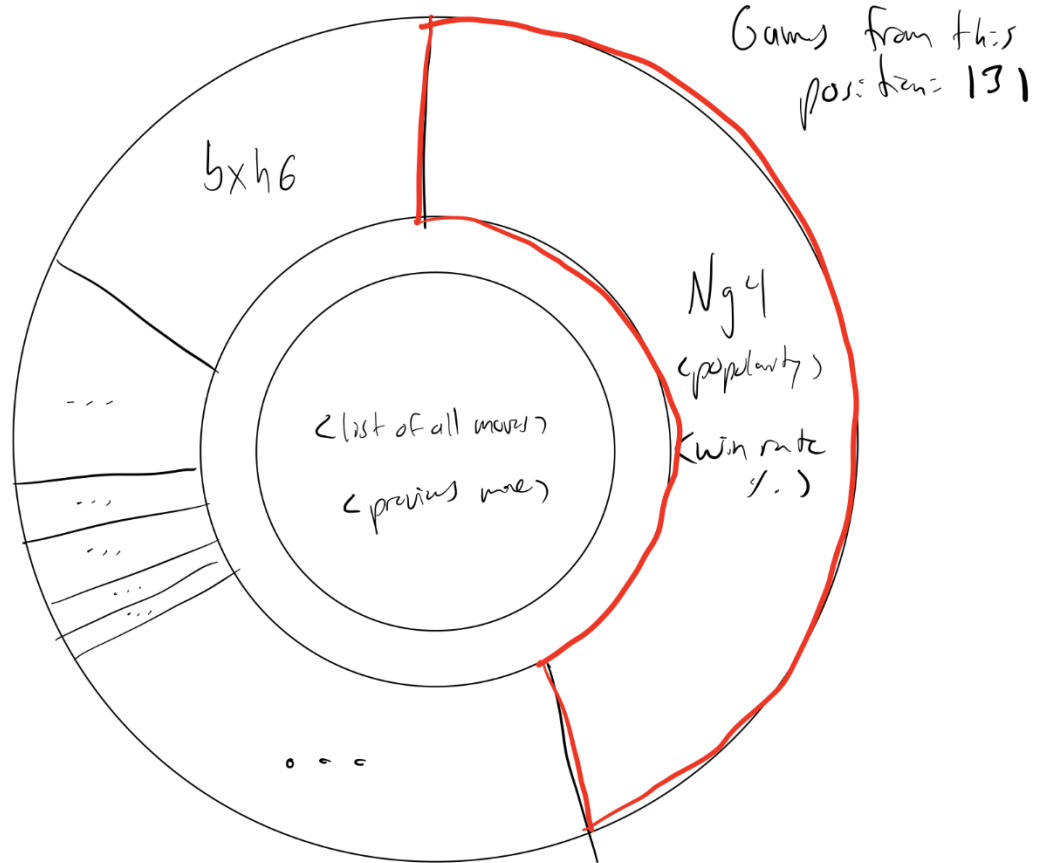
### Prototype 3

User can click on a wedge to expand it to the next ring.

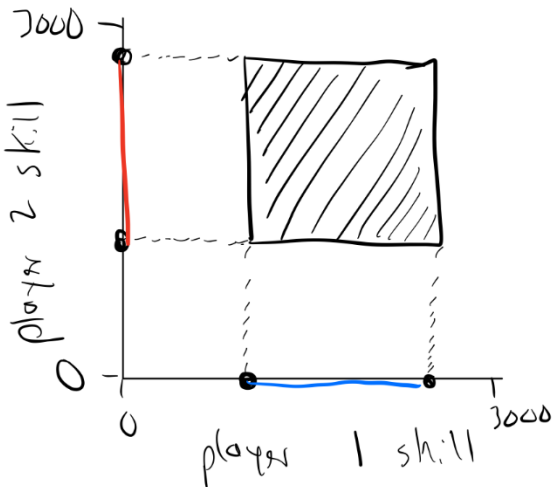




## Finalization



User can click on a wedge on it becomes the outer ring, or make a move on the chess board.



				X	X	
X	X	X	X			X
		X			X	
						X
					X	
X	X	X	X			

## Must-Have Features

1. A way to filter the skill level of the games in the dataset.
2. A way to see what moves are possible from a given position and the ability to select one for further analysis
3. Statistics about the selected move: usage rate, win rate, loss rate, draw rate, average rating of players that made this move, and number of games in dataset matching the selected path.

## Optional Features

1. A chess board that displays the current state and allows the user to make moves, updating the rest of the visualization.
2. A resizable brush tool to select the acceptable rating range of the games in the dataset.
3. An entry box that take a game in PGN format and loads it into the visualization.

## Project Schedule

I don't have teammates, so all tasks will be done by me.

### **Week 10: Project peer feedback**

- Get data wrangling done (download, convert to JSON, calculate derived data)
- Figure out how to group data for display
- Figure out how many games can be loaded effectively

### **Week 11:**

- Get the basic pie chart display up
- Click a wedge to load a new chart

### **Week 12: Project milestone due**

- Get a manipulatable chess board loaded
- Extract the latest move from the chess board

### **Week 13: Project review**

- Connect chessboard and pie chart visualization together so they can update each other
- Add documentation to code base

### **Week 14:**

- Create a resizable brush tool to control filtration parameters
- Get transitions working smoothly
- Add information to project website

### **Week 15: Final project due**

- Complete the process book and add it to the website
- Implement entry box that takes PGN string and loads the game into the visualization
- Submit project