<u>How Can I Improve My Chess Rating?</u>

Alden Kallabat

Lichess.org is one of the most widely used online platforms for playing real-time chess matches against players from around the world. This platform uses a dynamic rating system to represent each player's skill level, helping to match opponents of similar ability and track progress over time. Currently rated 1320 on Lichess.org, my goal is to improve my rating as efficiently as possible, without investing in formal coaching or extensive study. To support this goal, I analyzed a dataset of over 20,000 Lichess.org matches to uncover patterns and strategies that can lead to measurable improvement. This project leverages data-driven analysis to identify practical, actionable insights for accelerating skill growth in online chess.

Hypothesis:

By analyzing trends in opening choices, time controls, and opponent rating ranges, I can discover actionable patterns that correlate with higher win rates and use them to improve my chess rating over time.

Part 1: Data Preparation in Excel

As a first step in this project, I sourced a dataset containing over 20,000 chess games from Mysar Ahmad Bhat on Kaggle.com. After downloading the file in CSV format, I used Excel to inspect and clean the data. This included standardizing column headers by removing special characters (except underscores), verifying that rows and columns were properly aligned, and ensuring consistent formatting. Using Excel's Power Query, I removed leading and trailing spaces, applied proper capitalization, and validated that each column adhered to its appropriate data type (e.g., boolean, integer, string).

Once I ensured that the data was clean and prepared for SQL, I exported the file and began my analysis.

Part 2: Data Exploration in SQL

To begin, I first created the database within PgAdmin 4 and prepared the proper columns with specific data types and conditions. Below is the SQL query used to prepare the table.

```
CREATE TABLE chess_data (
    game_id VARCHAR(50) PRIMARY KEY,
    rated BOOLEAN,
    turns INTEGER,
    victory_status VARCHAR(20),
    winner VARCHAR(10) CHECK (winner IN ('White', 'Black', 'Draw')),
    time_increment VARCHAR(10),
    white_id VARCHAR(50),
    white_rating INTEGER,
    black_id VARCHAR(50),
    black_rating INTEGER,
    moves TEXT,
    opening_code VARCHAR(10),
    opening_moves TEXT,
    opening_fullname TEXT,
    opening_shortname TEXT,
    opening_response TEXT,
    opening_variation TEXT
);
```

Exploratory Insights

While these analyses do not directly test the main hypothesis, they provide valuable context and uncover meaningful patterns within the dataset. These insights help deepen the understanding of rating dynamics and gameplay trends, offering additional guidance for strategic improvement.

A)

What color chess pieces typically win more?

- This would help me indicate whether I am at a disadvantage early in the chess match.

	winner character varying (10)	number_of_wins bigint
1	White	5459
2	Black	5089
3	Draw	453

- The query above returned the total number of wins for White, wins for Black, and draws, based on the six game modes with the highest average player ratings and at least 500 recorded games each.

B)

What is the average skill (rating) difference between each player?

- This would help me understand what the typical rating difference is per time control.

	time_increment character varying (10)	total_games bigint	avg_rating_gap numeric	overall_avg_rating_gap numeric
1	5+8	697	199	162
2	5+5	738	180	162
3	15+0	1311	168	162
4	10+0	7721	163	162
5	8+0	588	141	162
6	15+15	850	120	162

- This query calculates total games and average rating gaps for selected time controls. The main query adds the overall average rating gap and sorts by gap size.

Hypothesis Testing

1. What are the most successful openings to play as white, and as black?
    - This will help me select an opening for both White and Black pieces that maximizes my chances of winning.

	opening_shortname text	total_games bigint	white_wins bigint	black_wins bigint	white_win_percent numeric	black_win_percent numeric	opening_initiator text
1	Philidor Defense	663	396	267	60	40	Black
2	Queen's Gambit	877	512	365	58	42	White
3	English Opening	691	395	296	57	43	White
4	Ruy Lopez	809	451	358	56	44	White
5	Scandinavian Defense	690	358	332	52	48	Black
6	Italian Game	934	483	451	52	48	White
7	Caro-Kann Defense	563	294	269	52	48	Black
8	French Defense	1342	689	653	51	49	Black
9	King's Pawn Game	881	440	441	50	50	White
10	Queen's Pawn Game	1172	570	602	49	51	White
11	Sicilian Defense	2502	1203	1299	48	52	Black

- This table lists the 11 most popular openings with over 500 rated games recorded, showing the win percentages for both White and Black. I also indicated which side initiates the opening.

Conclusion:

Using the Queen's Gambit, English Opening, and Ruy Lopez as White can boost my win rate by over 8%, while playing the Sicilian Defense, French Defense, and Caro-Kann Defense as Black can increase my chances of winning by more than 6%.

2. Are there any specific time increment categories (game modes) that have an overall higher rating per player than other categories?
    - This would allow me to focus on categories (game modes) that generally produce higher rated players.

	time_increment character varying (10)	total_games bigint	avg_white_rating numeric	avg_black_rating numeric	avg_total_rating numeric
1	8+0	506	1707	1685	1696
2	5+5	570	1672	1654	1663
3	10+0	6817	1618	1620	1619
4	15+0	961	1587	1584	1586
5	5+8	523	1530	1505	1518
6	15+15	722	1477	1480	1479

- The query above generated a table showing the average player ratings for games played as White, as Black, and overall, grouped by game mode. Initially, I encountered an error related to the white_rating and black_rating columns on lines 4, 5, and 6. After investigating, I found that these columns were the wrong data type and needed to be converted to a numeric data type. Casting them to numeric resolved the issue and allowed the query to run successfully. To ensure meaningful insights, I also filtered the results to include only game modes with at least 500 recorded games, maintaining a sufficient sample size for analysis.

Conclusion:

By prioritizing time controls like 8+0, 5+5, and 10+0 I can maintain a rating that is 15% higher than the lower performing time controls, assuming consistent, average play.

3. What are the chances of winning against opponents ranging from 500 points lower to 500 points higher in 30-point intervals, and which rating difference yields the greatest rating gain over 100 games?

- Lichess.org offers filters that let me choose the rating range of my opponents, allowing me to decide whether it's more beneficial to play against significantly higher-rated players or those closer to, or below, my skill level. Since the Elo rating system awards more points for wins against stronger opponents and penalizes losses to weaker ones more heavily, these choices directly impact how efficiently I can improve my rating.

Part 1: Grouping rating differences every 30 points and calculating W/D/L percentage

	rating_diff numeric	total_games bigint	higher_win_percent numeric	draw_percent numeric	lower_win_percent numeric
1	-510	15	73.33	0.00	26.67
2	-480	103	83.50	2.91	13.59
3	-450	123	82.11	2.44	15.45
4	-420	122	71.31	2.46	26.23
5	-390	183	80.33	3.28	16.39
6	-360	215	81.40	2.79	15.81
7	-330	255	78.04	3.14	18.82
8	-300	307	72.64	2.61	24.76
9	-270	321	66.04	5.30	28.66
10	-240	389	70.44	5.66	23.91
11	-210	502	67.73	3.98	28.29
12	-180	541	64.14	5.55	30.31
13	-150	645	62.95	4.96	32.09
14	-120	795	56.35	5.41	38.24
15	-90	964	53.11	5.91	40.98
16	-60	1248	52.08	6.41	41.51
17	-30	1491	46.81	5.16	48.02
18	30	1516	50.26	5.67	44.06
19	60	1220	55.08	4.67	40.25
20	90	1046	58.32	5.35	36.33
21	120	859	62.98	4.77	32.25
22	150	693	64.65	4.33	31.02
23	180	607	65.90	3.79	30.31
24	210	507	72.39	3.75	23.87
25	240	413	73.85	5.08	21.07
26	270	365	76.71	3.84	19.45
27	300	309	74.11	4.53	21.36
28	330	269	78.44	1.12	20.45
29	360	220	78.18	3.64	18.18
30	390	178	80.90	3.93	15.17
31	420	172	81.40	3.49	15.12
32	450	142	82.39	7.75	9.86
33	480	131	80.15	6.87	12.98
34	510	17	88.24	0.00	11.76

- This query analyzes chess games by grouping rating differences into 30-point intervals, up to a 510 rating difference. It labels each game outcome as a higher-rated player win, draw, or lower-rated player win. Then it aggregates, counts, and calculates win/draw/loss percentages for each rating gap, showing how rating difference affects outcomes. I then created a virtual table for further analysis.

Part 2: Calculating rating change over 100 games using W/D/L percentages

	opponent_rating_diff numeric	total_games bigint	higher_win_percent numeric	draw_percent numeric	lower_win_percent numeric	expected_win_percent numeric	lower_expected_win_percentage numeric	rgw numeric	rgl numeric	rgd numeric	rcw numeric	rcl numeric	rcd numeric	rcw numeric	rcl numeric	rcd numeric	final_rating_change numeric
1	-510	15	73.33	0.00	26.67	95	5	1.0082	-18.9918	-8.9918	[null]	[null]	[null]	73.9344	-506.5102	0.0000	-433
2	-480	103	83.50	2.91	13.59	94	6	1.1870	-18.8130	-8.8130	[null]	[null]	[null]	99.1161	-255.6684	-25.6458	-182
3	-450	123	82.11	2.44	15.45	93	7	1.3952	-18.6048	-8.6048	[null]	[null]	[null]	114.5571	-287.4447	-20.9958	-194
4	-420	122	71.31	2.46	26.23	92	8	1.6366	-18.3634	-8.3634	[null]	[null]	[null]	116.7085	-481.6710	-20.5739	-386
5	-390	183	80.33	3.28	16.39	90	10	1.9156	-18.0844	-8.0844	[null]	[null]	[null]	153.8799	-296.4034	-26.5168	-169
6	-360	215	81.40	2.79	15.81	89	11	2.2363	-17.7637	-7.7637	[null]	[null]	[null]	182.0361	-280.8439	-21.6607	-120
7	-330	255	78.04	3.14	18.82	87	13	2.6030	-17.3970	-7.3970	[null]	[null]	[null]	203.1382	-327.4115	-23.2266	-147
8	-300	307	72.64	2.61	24.76	85	15	3.0196	-16.9804	-6.9804	[null]	[null]	[null]	219.3431	-420.4349	-18.2189	-219
9	-270	321	66.04	5.30	28.66	83	17	3.4895	-16.5105	-6.5105	[null]	[null]	[null]	230.4453	-473.1915	-34.5058	-277
10	-240	389	70.44	5.66	23.91	80	20	4.0152	-15.9848	-5.9848	[null]	[null]	[null]	282.8307	-382.1966	-33.8740	-133
11	-210	502	67.73	3.98	28.29	77	23	4.5981	-15.4019	-5.4019	[null]	[null]	[null]	311.4270	-435.7207	-21.4997	-146
12	-180	541	64.14	5.55	30.31	74	26	5.2378	-14.7622	-4.7622	[null]	[null]	[null]	335.9537	-447.4417	-26.4301	-138
13	-150	645	62.95	4.96	32.09	70	30	5.9323	-14.0677	-4.0677	[null]	[null]	[null]	373.4383	-451.4325	-20.1758	-98
14	-120	795	56.35	5.41	38.24	67	33	6.6772	-13.3228	-3.3228	[null]	[null]	[null]	376.2609	-509.4634	-17.9763	-151
15	-90	964	53.11	5.91	40.98	63	37	7.4660	-12.5340	-2.5340	[null]	[null]	[null]	396.5202	-513.6426	-14.9758	-132
16	-60	1248	52.08	6.41	41.51	59	41	8.2900	-11.7100	-1.7100	[null]	[null]	[null]	431.7446	-486.0810	-10.9609	-65
17	-30	1491	46.81	5.16	48.02	54	46	9.1387	-10.8613	-0.8613	[null]	[null]	[null]	427.7812	-521.5611	-4.4445	-98
18	30	1516	50.26	5.67	44.06	46	46	10.8613	-9.1387	0.8613	545.8904	-402.6498	4.8837	[null]	[null]	[null]	24
19	60	1220	55.08	4.67	40.25	41	41	11.7100	-8.2900	1.7100	644.9853	-333.6736	7.9856	[null]	[null]	[null]	23
20	90	1046	58.32	5.35	36.33	37	37	12.5340	-7.4660	2.5340	730.9818	-271.2404	13.5568	[null]	[null]	[null]	33
21	120	859	62.98	4.77	32.25	33	33	13.3228	-6.6772	3.3228	839.0692	-215.3401	15.8497	[null]	[null]	[null]	25
22	150	693	64.65	4.33	31.02	30	30	14.0677	-5.9323	4.0677	909.4768	-184.0199	17.6131	[null]	[null]	[null]	70
23	180	607	65.90	3.79	30.31	26	26	14.7622	-5.2378	4.7622	972.8277	-158.7583	18.0487	[null]	[null]	[null]	120
24	210	507	72.39	3.75	23.87	23	23	15.4019	-4.5981	5.4019	1114.9460	-109.7558	20.2573	[null]	[null]	[null]	55
25	240	413	73.85	5.08	21.07	20	20	15.9848	-4.0152	5.9848	1180.4775	-84.6003	30.4028	[null]	[null]	[null]	71
26	270	365	76.71	3.84	19.45	17	17	16.5105	-3.4895	6.5105	1266.5220	-67.8704	25.0004	[null]	[null]	[null]	78
27	300	309	74.11	4.53	21.36	15	15	16.9804	-3.0196	6.9804	1258.4181	-64.4985	31.6213	[null]	[null]	[null]	171
28	330	269	78.44	1.12	20.45	13	13	17.3970	-2.6030	7.3970	1364.6206	-53.2314	8.2846	[null]	[null]	[null]	160
29	360	220	78.18	3.64	18.18	11	11	17.7637	-2.2363	7.7637	1388.7649	-40.6562	28.2598	[null]	[null]	[null]	176
30	390	178	80.90	3.93	15.17	10	10	18.0844	-1.9156	8.0844	1463.0282	-29.0596	31.7717	[null]	[null]	[null]	151
31	420	172	81.40	3.49	15.12	8	8	18.3634	-1.6366	8.3634	1494.7778	-24.7459	29.1881	[null]	[null]	[null]	174
32	450	142	82.39	7.75	9.86	7	7	18.6048	-1.3952	8.6048	1532.8523	-13.7563	66.6875	[null]	[null]	[null]	135
33	480	131	80.15	6.87	12.98	6	6	18.8130	-1.1870	8.8130	1507.8604	-15.4075	60.5452	[null]	[null]	[null]	210
34	510	17	88.24	0.00	11.76	5	5	18.9918	-1.0082	8.9918	1675.8327	-11.8569	0.0000	[null]	[null]	[null]	134

- Elo Rating System: The ERS is a mathematical formula used to calculate the relative skill level of players in games like chess, esports, and other competitive environments. Specifically for chess, the rating of a player is determined by the following two equations:
    1) $E = 1 / (1 + 10^{(Rb-Ra/400)})$
        a) E is the expected outcome in the Elo formula, indicating the likelihood (as a percentage or probability) that the player with rating Ra will win
        b) Ra represents the main player's current rating prior to the match
        c) Rb represents the current rating of the opposing player prior to the match
    2) $Rnew=Rold+K \cdot (Score-E)$
        a) Rnew is the main player’s new total rating after the match is played
        b) Rold is the main player’s rating before the match is played
        c) K is 20, this is a constant
        d) Score is either 1 for a win, 0.5 for a draw, or 0 for a loss

Using the Elo formulas, you can calculate your new rating after a single match. This formula does not estimate rating based on multiple games played. To estimate the total rating change over 100 games, I adapted the formulas to incorporate the win, loss, and draw percentages that were previously calculated.

Elo Calculation:

1st Step (calculating expected results):
E = 1/(1+10^(rating_diff)/400)

2nd Step (calculating Rating Gain for wins, losses, and draws):
RGW = 20(1-E)
RGL = 20(0-E)
RGD = 20(0.5-E)

3rd Step (calculating Rating Change for wins, losses, and draws using percent chances from table):
RCwin = RGW(100xhigher_win_percent)
RCloss = RGL(100xlower_win_percent)
RCdraw = RGD(100xdraw_percent)

4th Step (adding all 3 scenarios for a final predicted rating change after 100 games):
FRC = (RCwin) + (RCloss) + (RCdraw)

At first glance, when looking at the final rating change column, it may seem that playing higher-rated opponents leads to significant rating gains. However, playing stronger opponents doesn't necessarily result in significantly higher rating gains.

The lower_expected_win_percentage column represents the predicted chance of winning, for the lower rated player, against an opponent based on the rating difference shown in the opponent_rating_diff column. However, the expected win rate does not match the actual win rate for most rating differences, and the difference between them becomes larger as the rating difference increases.

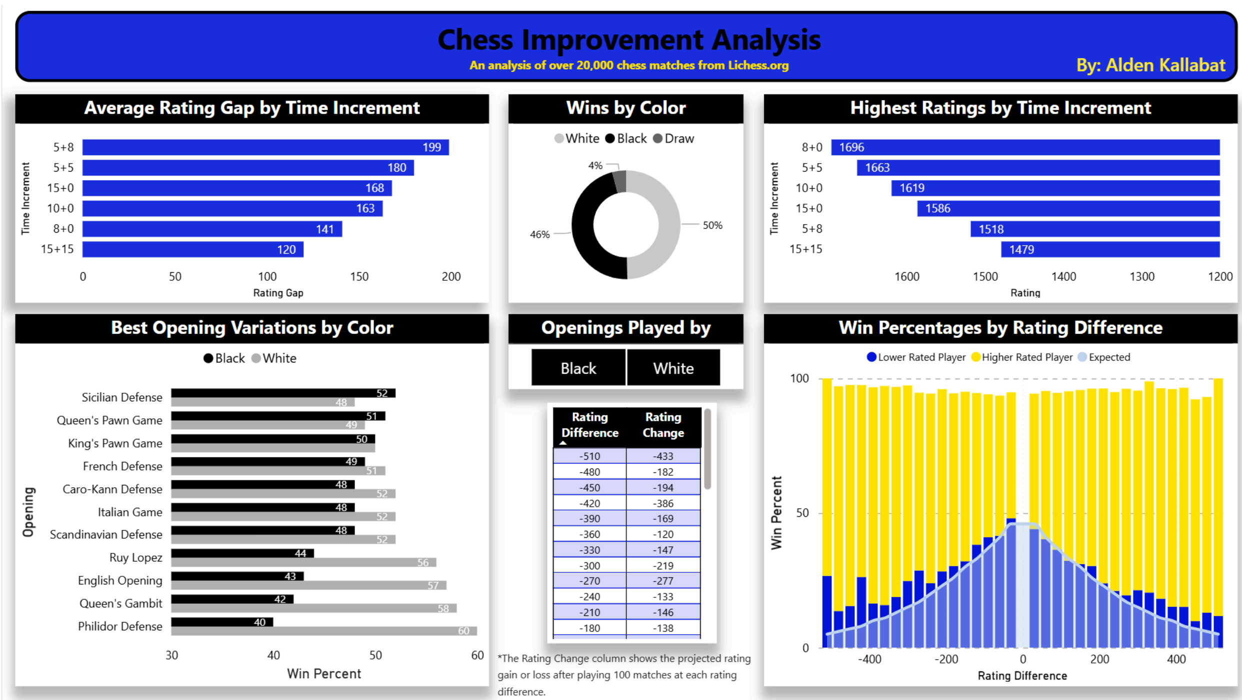
Based on research and experience, I believe this pattern is due to misclassification bias in player ratings. In online chess, it's easy to lower your rating but difficult to raise it. It's common for strong players to create lower-rated alternate accounts to avoid risking their high ratings on their main accounts. However, the reverse is not possible: lower-rated players cannot create new accounts that start with a high rating. Also, there are chess cheaters who use AI chess bots to provide them with the best move. These cheaters typically create new accounts and face higher rated players in order to rapidly increase their ratings. I could test this by analyzing rating history and assigning players an "authenticity" score based on game count and performance consistency, but Lichess.com does not release this data. Lichess addressed this issue by adopting the Glicko-2 rating system, which improves on rating-scaling by factoring in rating uncertainty and volatility. However, it still does not fully account for cheating and long-term performance consistency.

Conclusion:
Based on the available data, it appears unlikely that a definitive rating difference consistently yields the highest rating gains over a 100-game span. However, the analysis suggests that

playing opponents roughly 180 rating points higher offers the most effective balance between risk and reward, producing 170% more rating gains when compared to playing opponents 150 points higher, while the percent chance of losing increases by only 2%. Also, playing at a maximum rating difference of 180 should minimize the chance of playing against cheaters and strong players on new accounts.

Hypothesis Conclusion:
By analyzing and implementing trends in opening choices, time controls, and opponent rating ranges, I can improve my chess rating by up to 51%, bringing my initial rating of 1320 up to 1993 after completing 100 matches. This gain includes a 14% boost from using optimal openings, a 28% increase by consistently playing the 8+0 time control, and a 9% improvement from targeting opponents roughly 180 points higher in rating.

Power BI Dashboard:



- Slicers for Black and White openings are included
- Rating Difference vs Rating Change table has a scroll bar to view all rating differences
- Selecting a Rating Difference in either the Rating Change table or the Win Percentages by Rating Difference chart filters the other to that selected rating difference.

Resources

Bhat, M. A. (n.d.). *Online Chess Games* [Data set]. Kaggle.
https://www.kaggle.com/datasets/mysarahmadbhat/online-chess-games