Project 1 - Data Feature Extraction and Visualization

Group 418

Josiah L. Plett - s241748

Trevor G. Du - s241722

Tonglei Liu - s233213

02450 - Intro to Machine Learning

September 24th, 2024

Group Responsibilities

Section	Trevor	Josiah	Tonglei
Dataset Description	20%	**80%**	0%
Detailed Data Explanation	0%	50%	50%
Data Visualization	40%	20%	**60%**
Discussion	**80%**	20%	0%
Exam Problems	25%	25%	50%

Dataset Description

Dataset summary

Our dataset is the lichess.org public database of chess variant games, specifically Chess960 — or Fischer Random — games, which can be found here ("Lichess.org Open Database"). We are classifying and performing PCA on the 265,504 Chess960 games played in August 2024.

The games are stored in PGN format, which contains game metadata like the ratings of both parties, the randomized starting position, and the game outcome, along with all the moves of the game notated with how long that move took to be played ("Portable Game Notation").

Problem of Interest

Any chess player who's played online will identify with the constant struggle of not knowing when to spend time thinking, and when to move fast. In this report we will start with PCA on the general details of the games, to get a better understanding of what attributes of a game's play impact the length of the game, which will in turn affect time usage.

Furthermore, in our second report with in-depth classification and regression, we will use the style of time usage in each game by each player to hopefully predict game outcomes better than rating differences alone.

Data Transformation and Learning Goals

Our goal is to shine a light on how exactly Chess960 players — and by plausible extension, standard chess players — should plan to use their time in online rated games, to maximize their chances of winning.

The major steps in our analysis that will hopefully take us to such insightful conclusions are roughly the following.

1. Remove all non-blitz games: anything faster than 3|0 and slower than 5|5 (resource).
2. Compute at what points the middlegame and endgame started.
3. Perform PCA on the middlegame and endgame points, the length of the game, and the outcome of the game, with respect to the elo discrepancy between the players.

Looking forward to our *next* report, we'll be classifying average move time in different periods of the game as well as variance between move times. We'll use regression to predict the likelihood of a win, which will give us insight into what styles of time usage are most likely to lead to a win.

Previous Analyses of Move Times and Chess960 games

In this article by Emir U., an experienced Data Scientist, he explores exactly how much total thinking time affects players' performance. He matched up individual players' long games and short games, and used their comparative performance against identical opponents to make a profile of which players were faster thinkers. His conclusion was simply that yes, some people think faster than others.

In this article by Welyab Paula, an AI and Database Engineer, he investigates the win rates of chess960 games as a function of the initial board setup. He concluded that most boards are advantageous to white, and very few are advantages to black. The fact that any are advantageous to black is intriguing, and is relevant to our analysis. Unfortunately, the computation to account for this variable is outside the scope of this Machine Learning report.

Detailed Explanation of the Data Attributes

Overall Data Summary

Our full, modified dataset contains fourteen attributes: *Game result, White name, Black name, White elo, Black elo, Elo difference, Time control, Termination reason, Initial board position, Total moves, White move times, Black move times, Middlegame,* and *Endgame*.

Six of these attributes were computed by us from the initial dataset: *Elo difference, Total moves, White move times, Black move times, Middlegame,* and *Endgame*. *Elo Difference* and *Total moves* were simple, but the other four were tricky. We got the *White/Black move times* by parsing the PGN of the full game and translating it into a simple array of seconds. For *Middlegame* and *Endgame*, we classified those points in the game using the same methods lichess.org does, adjusted for better calibration with Chess960 (see the comments in our `pgn_parser.py isMiddlegame` and `isEndgame` functions if you'd like details).

Data Attribute Detailed Description

The following are the seven most practical-to-analyze values we performed PCA on.

WhiteElo: Continuous + Ratio variable. Normal distribution. Chess960 elo of the white player.

BlackElo: Continuous + Ratio variable, Normal distribution. Chess960 elo of the black player.

EloDifference: Continuous + Interval variable. Normal distribution. Represents the absolute difference between WhiteElo and BlackElo.

Middlegame: Continuous + Ordinal variable. Normal distribution. Represents the number of plies, or half-moves, played before the middlegame phase was entered. A -1 value means the middlegame was never reached; excluded when analyzing.

Endgame: Continuous + Ordinal variable. Normal distribution. Represents the number of plies, or half-moves, played before the endgame phase was entered. A -1 value means the endgame was never reached; excluded when analyzing.

TotalMoves: Continuous + Ratio variable. Right-skewed distribution. Represents the duration of the game in plies, or half-moves.

Result: Discrete variable + Nominal. Its three categories, in decreasing frequency, are White Win (1-0), Black Win (0-1), and Draw (½-½).

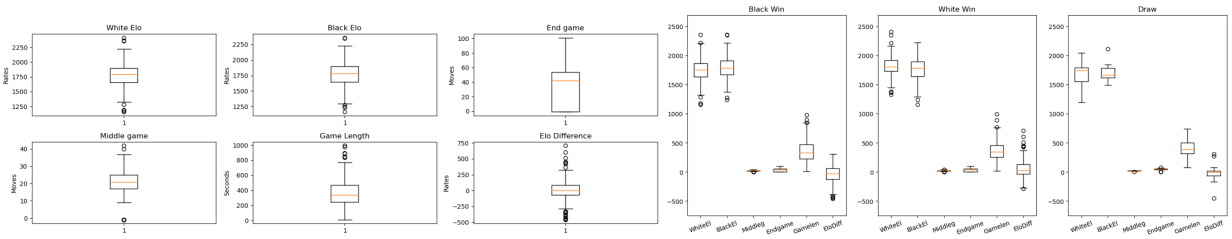
Data Issues

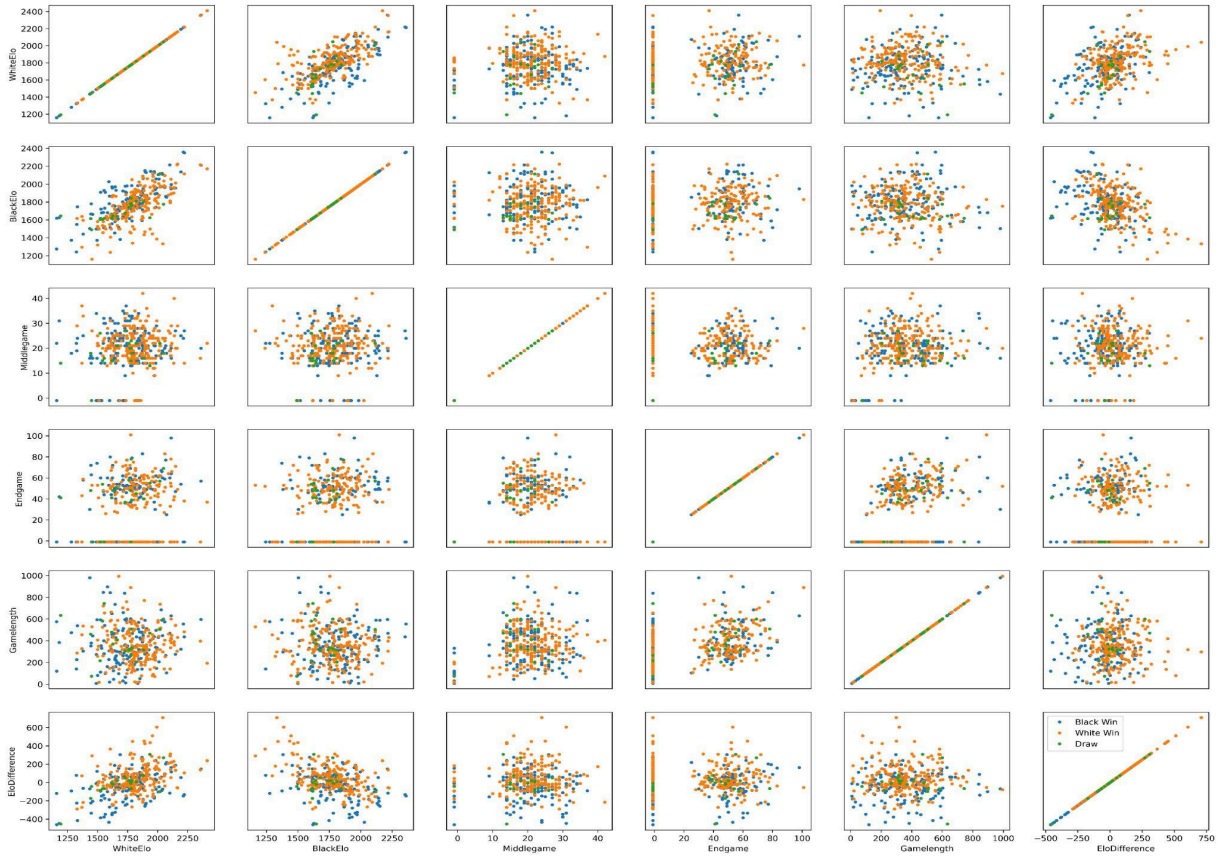
One issue is that our values are not normalized, so some values may be over or underrepresented compared to others. Another is that *Middlegame* and *Endgame* involve a -1 sentinel value for games that don't reach the corresponding phase; as a result, these attributes have to be analyzed independently with all -1 values removed to avoid swaths of problematic outliers both skewing the analysis and forcing the values to be considered Ordinal instead of Interval. This is essentially missing data because we do not know when the *Middlegame* or *Endgame* begin in these games.

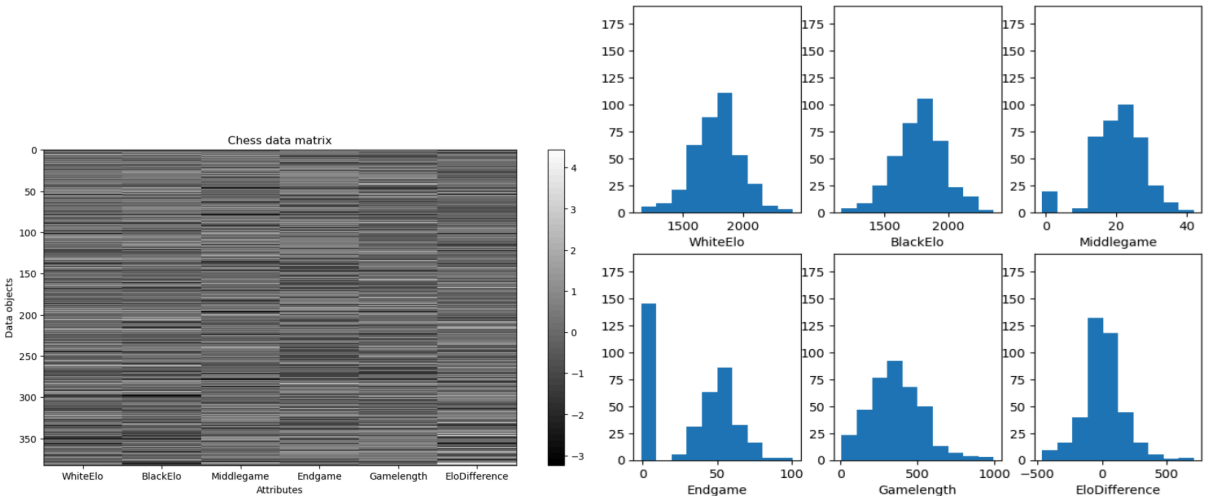
Data visualization

Miscellaneous Visualizations

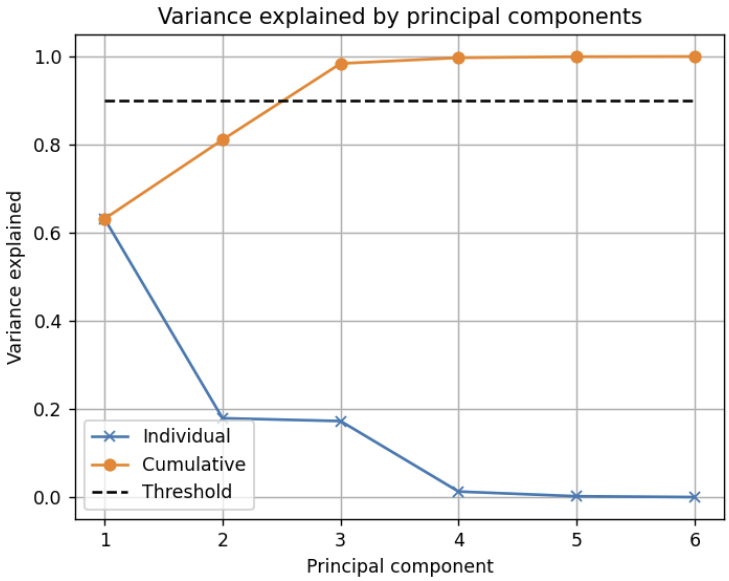
The following graphs are various visualizations to support a broader understanding of our data set.



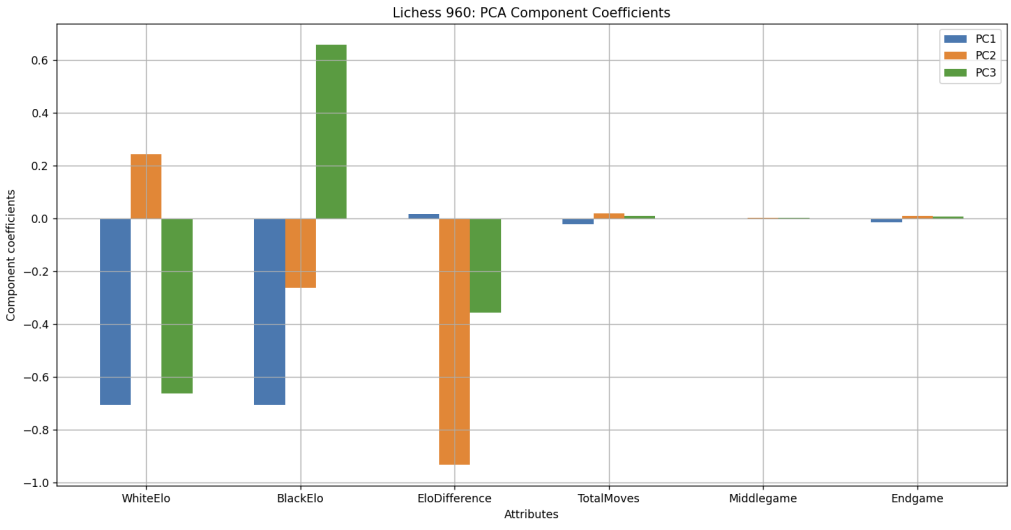

WhiteElo
BlackElo
Middlegame
Endgame
Gamelength
EloDifference
Black Win
White Win
Draw


Chess data matrix
Data objects
Attributes
WhiteElo
BlackElo
Middlegame
Endgame
Gamelength
EloDifference

Principal Component Analysis Results



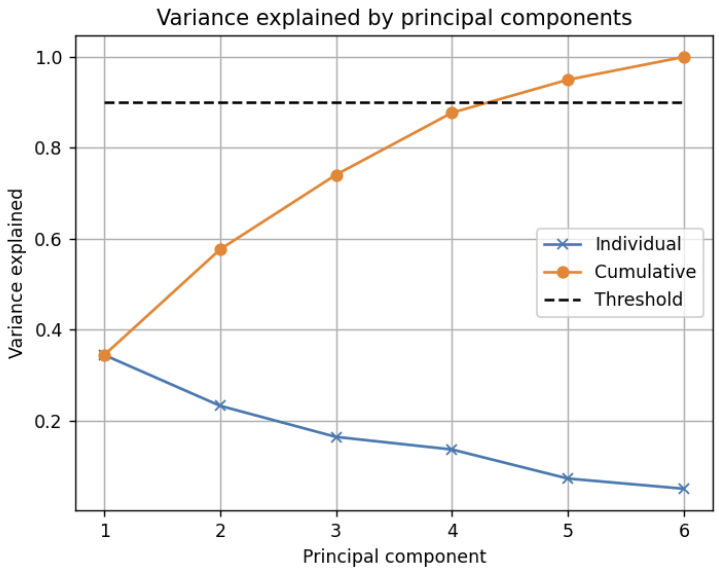
With a visual reference line (threshold) at ρ = 90%, we see that the first three principal components cumulatively account for over 95% of variance in the dataset. Below, we provide the component coefficients for the first three principal components.

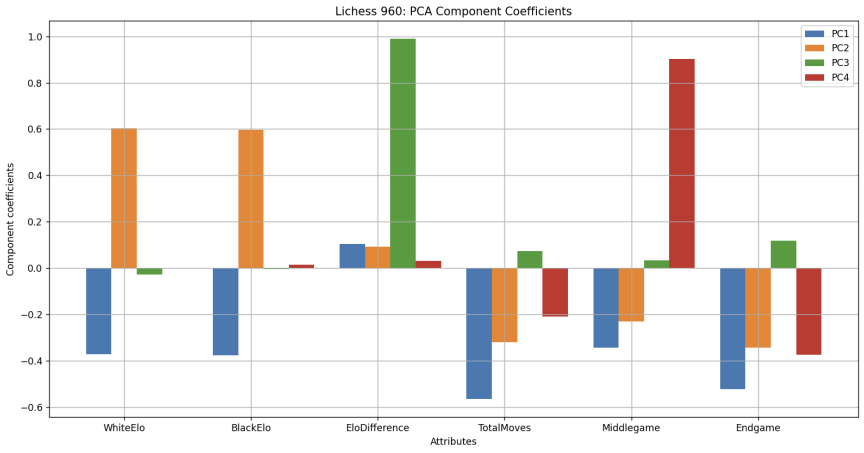


Here, we notice in the first three principal components that *WhiteElo, BlackElo,* and *EloDifference* are highly prioritized while the other three attributes are nearly completely disregarded.

Principal Component Analysis - Second Attempt

This time we normalize all attributes by subtracting their means and dividing by their standard deviations. Previous results were biased towards elo numbers since those attributes have the greatest magnitude and were thus overrepresented in our principal component analysis.





With four principal components, we can represent approximately 90% of the variance of *Result*. This time, the other attributes have gained greater representation as opposed to the principal components being dominated by *WhiteElo, BlackElo,* and *EloDifference*.

Discussion

We have learned a great deal about our data over the course of this project. First, we constructed a preliminary dataset on Chess960 games, identifying potentially useful metrics such as the beginning of each phase of the chess game (start, middle, end). Next, we discovered and took note of issues within our data. For example, there exist missing values for the *Middlegame* and *Endgame* attributes. From there, we visualized the distribution and spread of each attribute within the data. Lastly, we decided on seven attributes to analyze using PCA.

We noticed that out of these seven attributes, all of them are approximately normally distributed with the exception of *TotalMoves*, which is right-skewed. It is also easy to see our missing data problem in *Middlegame* and *Endgame* by noticing the spikes of outliers at -1, the sentinel value, in each of their distributions. From there, we provide scatterplots of the dataset with respect to two attributes, for each pair of attributes in the data set. We note some correlation between *WhiteElo* and *BlackElo*, which is simply a product of the Lichess rating system's goal to match together players of similar skill level. This bias is only a minor concern because, in general, only players of similar elo play interesting games against each other anyway.

Finally, we perform PCA on our seven selected attributes with respect to the *Result* attribute. As discussed previously, the almost sole source of variation in *Result* is explained by *WhiteElo, BlackElo,* and *EloDifference*. Using PCA, we could feasibly reduce the dimensionality of each data point from 6 to 4, while retaining 90% of the original variance of the *Result* attribute. A 10% loss in information could be deemed to be worth the 33% reduction in input size. It is clear that our task is a feasible machine learning task because our data has relatively few issues, the biases that exist are low-impact, and our chosen metrics do well to predict variations in a game's result.

Exam Problem Solutions

Question 2

Answer: **Option A.**

First step, find vector $x_{diff} = x_{14} - x_{18} = [7,\ 0,\ 2,\ 0,\ 0,\ 0,\ 0]$

Now let us simply go through each option.

A. $d_{p=\infty}(x_{14}, x_{18}) = max(|x_{diff,1}|, \ldots, |x_{diff,7}|) = max(7,\ 0,\ 2,\ 0,\ 0,\ 0) = 7$

We can stop here since our answer for **A** matches. Option **A** is correct.

Question 3

Answer: **Option A.** V (the first four components) =(s1^2 + s2^2+ s3^2+ s4^2) /(s1^2 + s2^2+ s3^2+ s4^2+ s5^2) = 0.866.

Question 4

Answer: **Option D.** In component 2, factors with high values have positive weights, while low values have negative weights. Therefore, they result in a positive sum.

Question 5

Answer: **Option A.** We can compute the Jaccard similarity using the equation J(x, y) = f11 / (K − f00). K is the total number of attributes so it's 20000 in our case. The number of words contained in both sets is 2, and the number of words contained in neither set is 19987. This gives a Jaccard similarity of 2 / (20000 - 19987) = 0.15385

References

"Lichess.org Open Database." *Lichess.org Open Database*, 1 Sept. 2024, https://database.lichess.org/#standard_games.

"Portable Game Notation." *Wikipedia*, Wikimedia Foundation, 5 Sept. 2024, en.wikipedia.org/wiki/Portable_Game_Notation.