Project 2 - Supervised Learning: Classification and Regression

Group 418

Josiah L. Plett - s241748

Trevor G. Du - s241722

Tonglei Liu - s233213

02450 - Intro to Machine Learning

October 27th, 2024

Group Responsibilities

Section	Trevor	Josiah	Tonglei
Abstract	0%	**100%**	0%
Regression - Part A	0%	**100%**	0%
Regression - Part B	0%	15%	**85%**
Classification	**85%**	15%	0%
Discussion	30%	**40%**	30%
Exam Problems	25%	**50%**	25%

Abstract

The dataset we're using is the lichess.org public database of chess variant games, specifically Chess960. All attributes in this report were computed by us, by analyzing the PGNs of each game. We also filtered out non-3|0 games, resulting in **45,343** games from August 2024.

The goal of this report is to explore how game length is related to clock management, and by extension, whether the final outcome can be more than just the difference in elo. We're doing this by predicting ***Total Plies*** with regression, and game ***Result*** with classification.

Regression - Part A

1 - Transformations and Prediction Variable

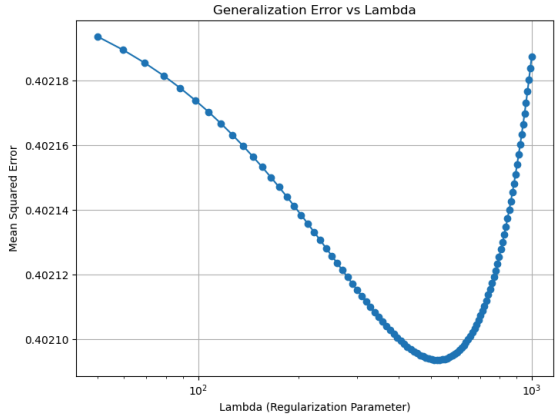
We are predicting ***Total Plies*** based on other attributes that are related to game length. They are: ***Elo Difference*** *(unsigned)*, ***Middlegame***, ***White Opening Time***, ***Black Opening Time***, ***White Total Time***, and ***Black Total Time***.

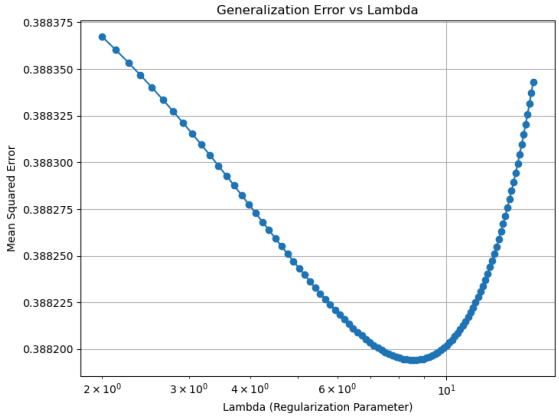
Elo Difference is unsigned because the difference in skill matters more in game length, not how much better White is than Black. *Middlegame* is the ply where the middlegame began. Attributes ending in *Time* are the percentage of total time (180 seconds) that was used during that period, between 0 and 1. *Termination* is nominal, either "Normal" or "Time forfeit." We have applied a feature transformation to the continuous attributes — which is all of them since we intentionally excluded *Termination* — to set their mean to 0 and standard deviation to 1.

We hope to predict *Total Plies* with better accuracy than simply looking at *Total Time*, which would show that the other attributes we've included, such as time usage in the opening, have statistically significant effects on the length of the game.

2 - Generalization Error

We experimented with different ranges of regularization parameter λ being applied to our basic Linear Regression model and assessed them via 10-fold cross-validation, until we found the global minima and its corresponding λ. The first graph is the result when using our full 45,343-game standardized data set, and the second is when using a random 500-game subset.





It's immediately apparent that the change in Mean Squared Error is miniscule at these global minima. In fact, across the range $-10^{-6} < \lambda < 10^{1.5}$ on both datasets, the Mean Squared Error never deviates by more than 0.001.

Due to our highly variable yet undoubtedly correlated game data, it makes sense that regularization doesn't have a big effect. Linear regression relies on plain old probabilities more than other models, so given how linear in nature our data is, "punishing overfitting" isn't really very punishing. Put simply, linear regression is so good at its job, regularization can barely help.

3 - Error Minimization Discussion

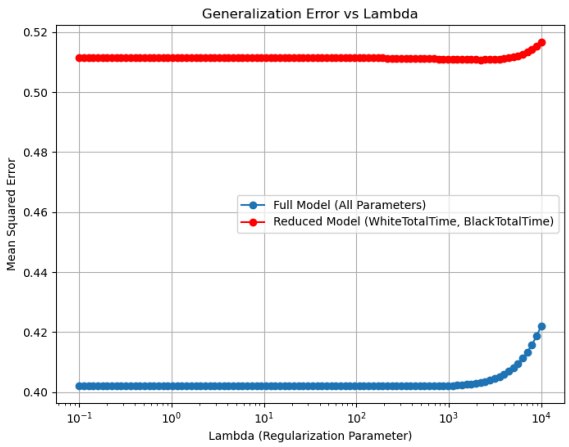
The linear model trained with optimal $\lambda = 520.2$ computes *Total Plies* with the following:

Parameter	**Coefficient**	**Aligns with intuition?**
Elo Difference	0.0240	No
Middlegame	0.2011	Unsure
White Opening Time	-0.2410	Yes
Black Opening Time	-0.2431	Yes
White Total Time	0.3996	Yes
Black Total Time	0.4686	Unsure

Let's discuss in increasing order of interest. The *Opening Time* parameters have a negative correlation with *Total Plies*. Since **34.14%** of games in our dataset end in time forfeits, using more time in the opening clearly indicates a shorter full game. The *Total Time* parameters are also exactly as you'd expect; however it's curious that Black's time matters more than White's. White wins **3.04%** more games than Black in our dataset, so given that long games lead to time scrambles, Black's time would have a slightly bigger impact than White's.

Now, why does a *Middlegame* with more moves lead to a total game with more moves? This would mean the game is moving slower, and slower-paced games need more moves before a winner is decided. Lastly, a larger Elo Difference somehow indicates a (slightly) longer game! You would think bigger skill differences would lead to quicker games, but that's not so. The most realistic interpretation is that the higher rated player wants to reduce any potential variance in game outcome, so they'll play passively and slowly to squeeze small advantages out of the weaker player, leading to marginally longer games on average. That's pretty insightful!

To conclude, we should answer, "Did this model with six parameters really perform better than if it only had *Total Time* to learn from?" Yes, over 20% better when comparing Mean Squared error; see the graph to the right. This means the *Total Plies* of a chess game is informed by data other than just how long the game took to play. Blitz chess enthusiasts should take note.



Regression - Part B

1 - Two-level Cross Validation Summary

In this section, a two-layer cross-validation method will be used to evaluate the performance of three models in this project: the Artificial Neural Network (ANN) model, the Ridge Regression model, and the Linear Regression model without any features.

In the two-layer cross-validation process, both the outer and inner folds are set to 5, and the loss function used is the Mean Squared Error (MSE) Loss.The model is trained on the

inner training sets to obtain its hyperparameters. These are then validated using the inner validation set to compute the error. Within an outer fold's training set, the errors obtained from the 5 inner folds are compared, and the hyperparameters with smallest error are selected as the best hyperparameters to represent that training set. Across all outer folds, each training set's hyperparameters are used to evaluate the error in the outer fold's test set .The hyperparameters corresponding to the lowest final error are chosen as the optimal model hyperparameters.

Different values of hidden units will be treated as hyperparameters join the two-layer cross-validation process in the next phase. Considering the relatively small dataset in this project, hidden units will be tested in the range from 1 to 10. In the preliminary testing, each hidden unit value is selected as a specific number(from 1 to 10), and the error will be calculated. The top 40% of the hidden unit values with the smaller errors will then be selected and used in subsequent testing to improve efficiency which is 1,7,8,9,10.

The ridge regression coefficient λ will be selected from 0.1 to 1000 which follow a logarithmic uniform distribution without the preliminary test.

2 - Two-level Cross Validation Implementation

Table: Two-level cross-validation table used to compare the three models

Outer Fold	ANN		Ridge Regression		Baseline
i	hi	Ei	λi	Ei	Ei
1	10	0.346	0.1	0.394	0.449
2	8	0.408	0.1	0.462	0.394
3	9	0.341	2477.076	0.406	0.375
4	8	0.361	2782.559	0.381	0.376
5	7	0.387	0.1	0.372	0.415

The best λ is determined to be 0.1 using two-level cross-validation, which is completely different from the 520.2 given in the previous chapter.

By visualizing in the table, the error obtained for the most complex ANN model is slightly smaller but still similar to the other two models' errors. There is not an obvious difference between the ridge regression model and the baseline model. None of the three models seem to make much of a difference for this project.

All three models for this project perform similarly with negligible differences in error. Therefore, it can be concluded that the model complexity does not result in a significant performance improvement visually.

3 - Performance Comparison

Table: T-test for to compare the three models

Parameter	ANN vs Ridge	ANN vs Baseline	Baseline vs Ridge
p	$3.46 * 10^{-14}$	$3.46 * 10^{-14}$	0.45
CI(difference-z)	(-0.018, -0.011)	(-0.018 ,-0.010)	$(-8.263 * 10^{-8}, 3.719 * 10^{-8})$

Based on the p-values and confidence intervals, it is shown there is a significant performance difference between the ANN model and both the Baseline and Ridge Regression models. However, no significant difference was found between the Baseline and Ridge Regression models, suggesting that they perform similarly. Therefore, ANN shows a notable distinction in performance, while Baseline and Ridge Regression can be considered statistically identical in terms of model performance.

The ANN model performs the best while the Ridge Regression and Baseline models perform similarly.Since the ANN model performs better than both the baseline and ridge regression models, it is recommended to use, especially for tasks with complex, nonlinear relationships in the data.

Classification

1 - Classification Problem Summary

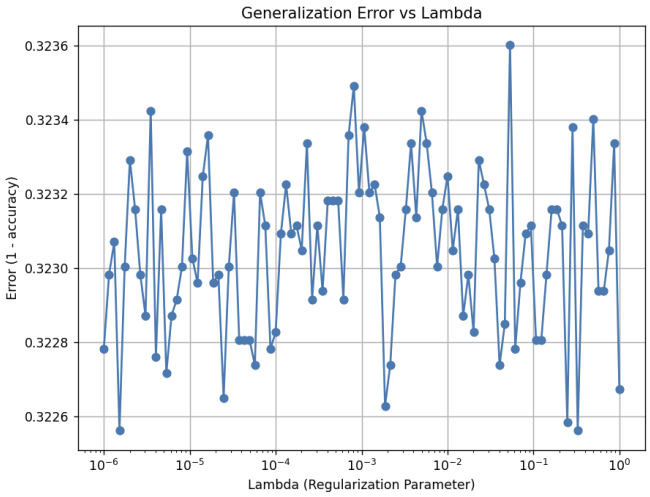
The relevant classification problem we'll be analyzing is classifying the multi-class **Game Result** attribute, which will be one-of-k encoded into *White Win, Black Win,* and *Draw*. Classification is for predicting Discrete variables, and with a dataset full of chess games, game outcome is the most natural attribute to predict.

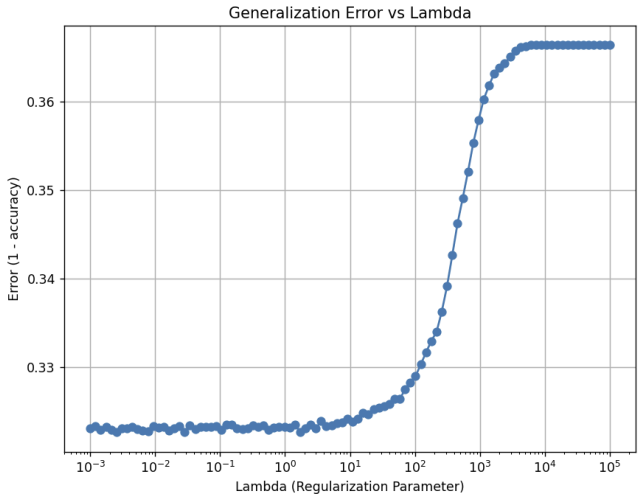
The inputs we'll be giving it are **Elo Difference** (unsigned), **Better Player** (binary, indicating whether white or black has a higher Elo), **Termination Reason** (one-of-k encoded), **Total Plies**, **Opening Plies + Middlegame Plies + Endgame Plies**, **White+Black Time in Opening+Middlegame+Endgame**.

For *method 2* we will be using **Classification Trees**. This is because when trying to predict **Game Result**, an ordinal attribute, it makes the most intuitive sense to find partitions in the inputted continuous variables that lead most directly to one prediction versus another.

2 - Complexity Controlling Parameter Selection

We once again experimented with a range of regularization parameters until finding a global minimum for error rate. Provided below are two plots of error with respect to λ.





It is interesting to note that it appears that λ close to zero are optimal for minimizing the error rate. Error rate increases logistically with larger values of λ until plateauing at 36.64% error, which exactly matches the error rate if nothing but the binary *Better Player* attribute is used for prediction. From the graph, we will examine the range 10^{-8} to 10^{0} for values of λ. Beyond 10^{0}, error begins increasing almost monotonically.

Error rate of the *Baseline* model which always predicts *White Win* is 51.12%.

3 - Model Comparison and Two-level Cross Validation

Outer Fold	Classification Trees		Logistic Regression		Baseline
i	$max\_depth^*_i$	E^{test}_i	λ^*_i	E^{test}_i	E^{test}_i
1	11	0.321389073	2.811768e-06	0.326034063	0.510285335
2	10	0.321167883	2.023589e-07	0.324264543	0.514709135
3	10	0.307232913	0.323745754	0.316522893	0.509621765
4	13	0.311435523	1.389495e-07	0.329130723	0.508073435
5	9	0.320725503	2.811768e-06	0.314310993	0.508737005
6	13	0.325370493	2.811768e-06	0.326255253	0.492590134
7	9	0.330679053	6.551285e-08	0.331342623	0.516921035
8	10	0.312099093	0.002442053	0.322495023	0.516921035
9	11	0.321610263	0.010985411	0.317850033	0.516036275
10	9	0.323379783	0.005179474	0.320504313	0.518248175

Depicted above is a tabulation of the results of each outer fold of a two-level cross validation procedure performed on classification trees, logistic regression, and baseline classifiers. On inner cross validation, we seek to find the optimal *max_depth* parameter for decision trees and the optimal *regularization parameter* for logistic regression. Notice that both decision trees and logistic regression have a significantly lower error rate than the baseline classifier, which means that our models are learning to draw meaningful conclusions from their input features.

4 - Statistical Evaluation

For this problem, we have chosen to use McNemar's test under **setup I** to evaluate and compare our models, resulting in the below table. Note that two layer cross-validation was used to generate optimal models per outer fold and use them to get predictions on test set data.

Parameter	Tree vs. Baseline	Regr. vs. Baseline	Tree vs. Regression
p	0.000	0.000	0.001916327747560651
CI (α = 0.5)	[0.18849505888512885, 0.2005708852447845]	[0.1819182883260071, 0.19409758708628955]	[0.002417567453837588, 0.010632534098027113]

Using a significance of α = 0.5, we see that both decision trees and regression clearly outperform the baseline model with a p-value of 0.000 and positive error difference (according to the confidence intervals). We can also see that there is a statistically significant difference in performance between the decision tree and regression models with a p-value of 0.0019.

5 - Logistic Regression

From the table we created in problem 3, we pick λ = 2.8117686979742253e-06 as the optimal regularization parameter value as it is the one that resulted in the lowest test set error during two-layer cross validation.

For binary classification problems, we can train a linear regression model and then apply a logistic function at the end to normalize its output between 0 and 1. This is a logistic regression model. We can convert a logistic regression model's output into a class by declaring 0.5 as a splitting point where all outputs below 0.5 correspond to one class and all outputs above 0.5 correspond to the other.

Our problem is a multi-class classification problem. We can easily extend our binomial logistic regression classifier into a multinomial logistic regression classifier by fitting one binomial logistic regression classifier for each possible class (black win, draw, white win). On prediction, we select the class with the model that has the largest output value.

We see that the input features for both the regression and classification parts of the project are largely the same, with additional input features being used by the classification part. For example, classification takes not only the opening times as input, but also the middlegame and endgame times. Classification applies some feature transformations that regression does not. The differences are overall minor. This is due to the fact that both problems seek to learn something about how a chess match will play out according to the players' elo differences and their respective management of the clock.

Discussion

Overall Discussion

In **Regression - Part A**, we analyzed the predictive ability of all time-based and game-period-based data to predict the length of the game in plies: *Total Plies*. Further, we explored how much a regularization parameter can increase our prediction accuracy. We concluded that the linear nature of our data lends itself well to a regression model and therefore resists overfitting, so our regularization parameter isn't very helpful. Furthermore, we discovered that using time spent in the opening and other metrics significantly increased the predictive power of our Regression model, indicating chess players can get a sense of how long their games will last based solely on time usage and opening length in the first part of the game.

In **Regression - Part B**, based on a two-fold cross-validation approach, we found the optimal hyperparameters for the Ridge Regression model (λ) and the ANN model (h). These optimized models were compared and included a baseline model. The results showed that the ANN model significantly performed better than the other two models which is the most suitable for this project. On the other hand, no significant difference was found between the Ridge Regression model and the baseline model, indicating that regularization has little to no effect in this project.

In **Classification**, we explored the limits of predicting *Game Result* based on our various continuous attributes surrounding time usage and pace of game, alongside the basics like Elo difference between the players. The vision was to see if a Classification Tree, a human-reproducible process for predicting attributes, can provide better insight into the game outcome than purely using Elo Difference (which alone gives a low 36.64% error). We concluded that it can, bringing total error down to around 30.7%. Meanwhile, our logistic regression model was able to bring error down to around 31.5% at its lowest. Due to how close these error measures were, it was unclear at first glance whether the difference in performance between classification trees and logistic regression is statistically significant. Using McNemar's test under setup I, we were able to determine, with a p-value of 0.0019, that classification trees do in fact perform better than logistic regression classifiers on a statistical level. We found that using formal methods to prove differences in model performance can result in much greater confidence in the models that we choose. Finally, we compared the input features passed to the

regression and classification parts of the project and found that they were not all that dissimilar. Primarily, this is due to the fact that both parts are using the same dataset. However, this just demonstrates how many different problems can be solved by recontextualizing the same set of data.

Previous Analysis of our Data

Our dataset, which we've pulled directly from the source — lichess variants database — and heavily processed, has not been previously analyzed. The closest thing is this repository that just statically counts the number of wins and losses for each board position. We considered using their results as further calibration of our data, but the effect would be too small to be insightful.

Exam Problem Solutions

Question 1

Answer: **Option C.** An ROC (Receiver Operator Characteristic) graph plots True Positive rate over False Positive rate, and in this problem, True Positives (y=1) are a red **X** and False Positives (y=0) are a black **O**. Given the graph, we should expect to find **True** Positives and **False** Positives in the following sequence:

1 ***True****, 2* ***False****, 2* ***True****, 2* ***False****, 1* ***True***

The only prediction plot that matches this pattern is Plot C.

Question 2

Answer: **Option C.** Since we're using *Classification Error* to measure impurity, we have:

$$CE_{node} = 1 - max\left\{\frac{n_{class}}{n_{total}}\right\}$$

Impurity gain is the measured impurity after a split minus before a split, so we have:

$$IG_{split} = CE_{root} - (n_{branch1} * CE_{branch1} + n_{branch2} * CE_{branch2}) / n_{total}$$

We can count $n_{total} = 37 + 31 + 33 + 34 = 135$. Now, the dominant class at the root has 37 items, so $CE_{root} = 1 - \frac{37}{135}$. Now we split on $x_7 = 2$, and define *branch1* as

$x_7 \neq 2$ and *branch2* as $x_7 = 2$. Then we have $n_{branch1} = 134$ and $CE_{branch1} = 1 - \frac{37}{134}$; and $n_{branch2} = 1$ and $CE_{branch2} = 1 - \frac{1}{1}$. This gives the final Impurity Gain equation:

$$IG_{x_7=2} = (1 - \frac{37}{135}) - (134 * (1 - \frac{37}{134}) + 1 * (1 - \frac{1}{1})) / 135 = \frac{1}{135} = 0.\overline{074}$$

Question 3

Answer: **Option A.** The structure of ANN is INPUT LAYER → HIDDEN LAYER → OUTPUT LAYER. HIDDEN LAYER with 10 units → 7 features * 10 units = 70 parameters ; OUTPUT LAYER : Amount of classes = 4, units of HIDDEN LAYER = 10 → 4 classes * 10 units = 40 parameters.Total parameter = 70+ 40 +4 + 10 = 124 = Option A

Question 4

Answer: **Option D.** We see that only one leaf node corresponds to congestion level 4: the child of C if its condition is true. In figure 4, the zone corresponding to congestion level 4 is divided from the other congestion zones only by a single value in b_1. Thus, it is immediately clear that the condition for C must be $b_1 \geq a$ where $-0.5 < a < 0$. The only option with such a condition for C is option D.

We had a great time building this report and finding truly new insights into a real-world dataset, despite it taking significant up-front effort. Thank you for this opportunity!