SkillCraft Multivariant Regression

Arrington Walters

11/4/2020

Introduction

Videogames are one of my favorite past times. As a player who participates in ranked play, I've always kept my eye on the forefront of global competitions. One of the most notable games to establish an international competitive scene backed by paid professionals was the real-time strategy game (RTS) Starcraft II (SC2). In 2013 the top 10 starcraft players made nearly four-million dollars from their combined winnings ("Winnings: 2019 - Liquipedia - the Starcraft Ii Encyclopedia" n.d.). Watching top-caliber players reflexes and control is astonishing to even seasoned videogame enthusiasts. At the 2019 StarCraft II World Championship Finale, many others and I packed into the arena to see what these professions could do firsthand.

Figure 1: '("Congrats to the Starcraft Ii Wcs Global Finals Champion! - Blizzcon" n.d.)'

The eye-watering speeds they perform at is universally referenced in gaming terminology as actions per minute (APMS). Professionals take actions at such fast speeds (high APMS); it becomes challenging to follow their overall strategy. Past pondering their sheer speed, I found it difficult to distinctly define what made these players high skilled.

To learn more about what defines talent in SC2 this analysis, we will explore in-game metrics to explain rank in competitive mode. The dataset used was provided by '("UCI Machine Learning Repository: SkillCraft1 Master Table Dataset Data Set" n.d.)'.

Goal

To model the response LeagueIndex a sample of player data from a 2013 ranked season of Starcraft will be explored. The predictors provided summarize in-game performance metrics for a season by player (GameID). The modeling process will consider all the predictor variables and then trim down until only significant predictors remain. Variables will be vetted for multicollinearity, and finally, the model will be explored to see if the BLUE assumptions hold.

The goal of this analysis will be to test the explanatory power of APMs and other predictors that are less commonly discussed.

Limitations of the Model

The multivariate regression model used for the midterm two portions of this study explores the linear estimation of mean response of LeagueIndex estimated by predictors in the design matrix X.

The assumptions of this model's explanatory power depend on the residual error being gaussian. Considering LeagueIndex is an ordinal variable, it is doubtful, if not impossible, for the residuals to be statistically normal.

A more suitable form of a model for this regression would be based on a Polytonomous Logistic Regression for Ordinal Response (Proportional Odds Model) ("Ordinal Logistic Regression | R Data Analysis Examples" n.d.). These methods will be revisited for the final portion of this analysis.

The Data Exploring

This dataset is a sample of averaged in-game metrics of Starcraft II players who participate in 2013 ranked play. The variables are as follows:

```
##  [1] "GameID"               "LeagueIndex"          "Age"
##  [4] "HoursPerWeek"         "TotalHours"           "APM"
##  [7] "SelectByHotkeys"      "AssignToHotkeys"      "UniqueHotkeys"
## [10] "MinimapAttacks"       "MinimapRightClicks"   "NumberOfPACs"
## [13] "GapBetweenPACs"       "ActionLatency"        "ActionsInPAC"
## [16] "TotalMapExplored"     "WorkersMade"          "UniqueUnitsMade"
## [19] "ComplexUnitsMade"     "ComplexAbilitiesUsed"
```

The appendix covers each in-depth, but the following are highlighted as a preface.

LeagueIndex The levels of LeagueIndex range 1-8 corresponding to player ranks Bronze, Silver, Gold, Platinum, Diamond, Master, Grand Master. Visible to the player in-game, each medal bronze through master is subdivided into divisions 1-5. Each division is once again but instead by divisions but is instead unbounded in terms of rank points ("Leagues: 2019 - Liquipedia - the Starcraft Ii Encyclopedia" n.d.). The rating system similar to an Elo rating system standard in chess. Elo's designs have an extreme value distribution, also known as a Gumbel distribution ("ELO Rating Sysem in Chess" n.d.). Although a Gumbel distribution would be problematic as a nonnormal response, it would provide some much-needed continuity by transforming players **LeagueIndex** into a more continuous experimental variable. Unfortunately, these subdivisions are either unavailable or would require far too much cleaning for the scope of this analysis.

The limitation of predicting this ordinal response will be revisited more precisely, along with the exploration, modeling, and predictions.

The following are the icon's earned for players who achieve related rank by the end of a given season. The legends for the following plots are styled to match.

Actions Per Minute (APMs) - APMs apply to various games but are the standard metric for analyzing proficiency of players at RTS games; its theorized skills like this provide a great advantage to players ("APM Definition" n.d.). Action quickness alone does not capture the strategy or macro/micro-skills, so these additional predictors may add some unique color in hopes of further explaining what makes players skilled.

Perception-Action Cycles (PACs) - are the circular flow of information between an organism and its environment where a sensor-guided sequence of behaviors is iteratively followed towards a goal ("Perception-Action Cycle - Models, Architectures, and Hardware | Vassilis Cutsuridis | Springer" n.d.). In this data-set, PACs are aggregate of screen movements where PAC is a screen fixation of containing at least one action ("UCI Machine Learning Repository: SkillCraft1 Master Table Dataset Data Set" n.d.).

Figure 2: '("Congrats to the Starcraft Ii Wcs Global Finals Champion! - Blizzcon" n.d.)'

Cleanliness

The missing values are related exclusively to players with LeagueIndex equivalent to Professional Players (8). The 55 players with LeagueIndex==8 the age data is NA and the HoursPerWeek are 0. LeagueIndexes 1-7 are obtainable playing matches in the base game and ranking up by winning. To be a professional, you would have to be part of a team that has no direct role in the broader matchmaking system. This study aims to understand how players go from being average to good, less so elite to best. The 55 values associated with professionals will be dropped to resolve both issues.

Another issue with **LeagueIndex** is that **LeagueIndex** 1-6 may contain many players, while **LeagueIndex** 7-Grandmaster may only include some set range of players targeted at 1000 total per region ("Leagues: 2019 - Liquipedia - the Starcraft Ii Encyclopedia" n.d.). Dropping **LeagueIndex**=7 would be a step towards normality. Considering this multivariate linear model is already hampered by its selected application on an ELO system, **LeagueIndex**=7 will be kept to preserve a potential insight into Starcraft II players' larger population.

In addition to the missing values we have a clear error with the **TotalHours** of one player. $GameID = 5140$ has 1,000,000 **TotalHours** that equates to 114 years of game time.

If we assumed one extra zero had been added at the end of the player's **TotalHours**, it equates to 14 years of playing time on a game that is only ten years old as of 2020. Removing two zeros equates to 1.4 years of playing time and three zeros in 51.1 days of played time, both that seem just as realistic. There is not a clear path to extrapolate this player's true **TotalHours**, so their data will be dropped from the analysis. This was initially detected during modeling but brought earlier into that analysis.

Finally, performing a necessary inspection on **HoursPerWeek**, a max value of 168 was discovered. Considering there are 168 hours in a week, it's not plausible for an individual player to do this. There could be multiple players using this account, making this possible. Another prospect is that this player is an AI like google's DeepMind ("AlphaStar: Mastering the Real-Time Strategy Game Starcraft Ii | Deepmind" n.d.). Either way, this observation will be kept because what is the realistic cutoff for hours per week is not apparent, and after removing this observation, the next max value is 140, which seems almost as unrealistic.

It's worth noting that dropping any amount of high hour outliers still far from combats all the potential abnormalities encountered through the use of **HoursPerWeek** and **TotalHours**. Multiple players could be using any of the accounts, even if either time played variable is not relatively large. Potentially exacerbating the left-extrema is that nothing prevents one player from smurfing multiple times. Smurfing is when a player makes an additional accounts ("What Is a Smurf Account? Everything You Need to Know | Lol-Smurfs" n.d.). A common reason for doing this is to dominate the competition until their Elo rating adapts to their actual skill level.

Converting Units

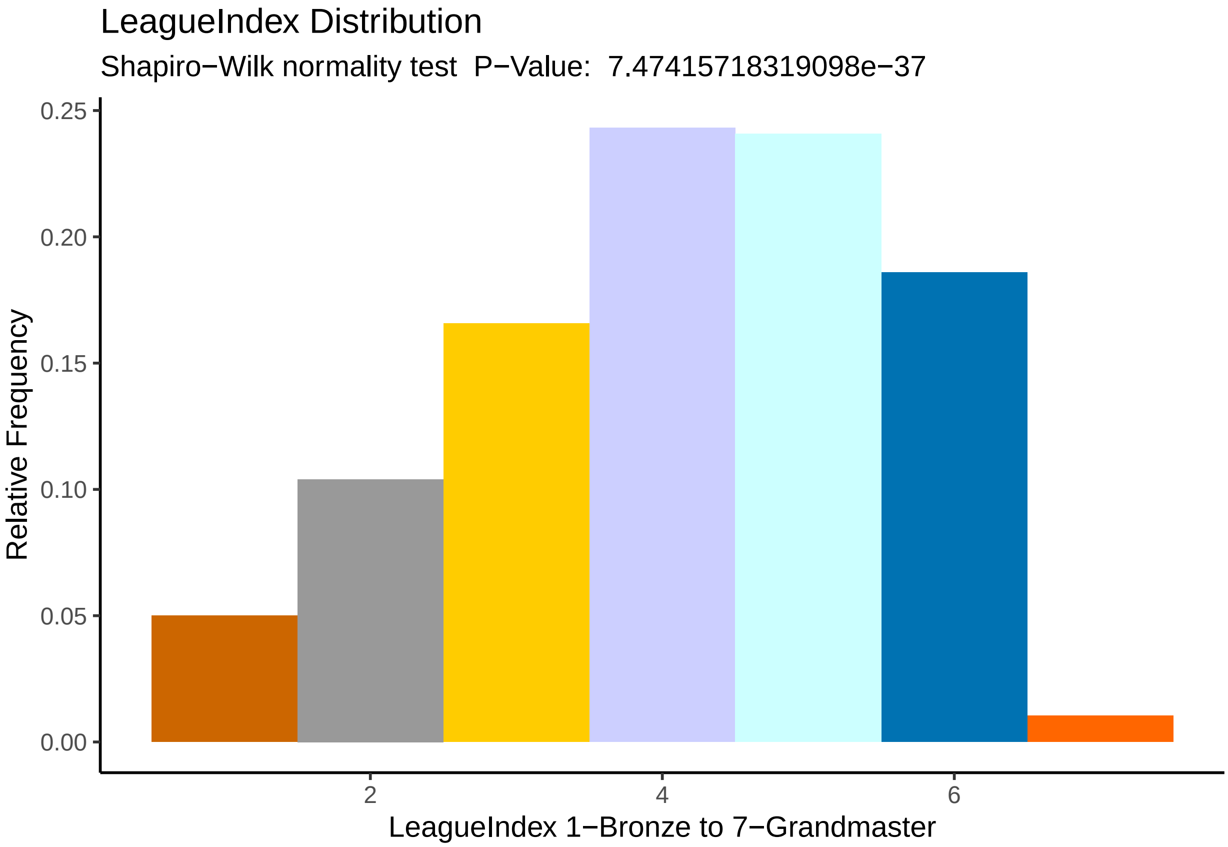
Some of the time, averaged metrics are per SC2 timestamp while others are per millisecond. All-time metrics in timestamps or milliseconds will be converted to seconds to help with interpretability. There are roughly 88.5 timestamps per second, so each metric in timestamps will be multiplied by 88.5 ("UCI Machine Learning

Repository: SkillCraft1 Master Table Dataset Data Set" n.d.). Some of the time-averaged metrics are per millisecond. This transformation is linear and will not affect our model's assumptions.

Summary Statistics and Plots

Gaussianity of the Response

When using the Shapiro-Wilk W test on response **LeagueIndex**, the null hypothesis that the sample comes from normally distribution can be rejected. Besides the obvious issues with performing a W test with an ordinal response with a potentially underlying Gumbel distribution, the response has a negative skew with a mean of 4.12. Furthermore, there is no reason that **LeagueIndexes** are uniforming spaced in terms of overall rank or skill.



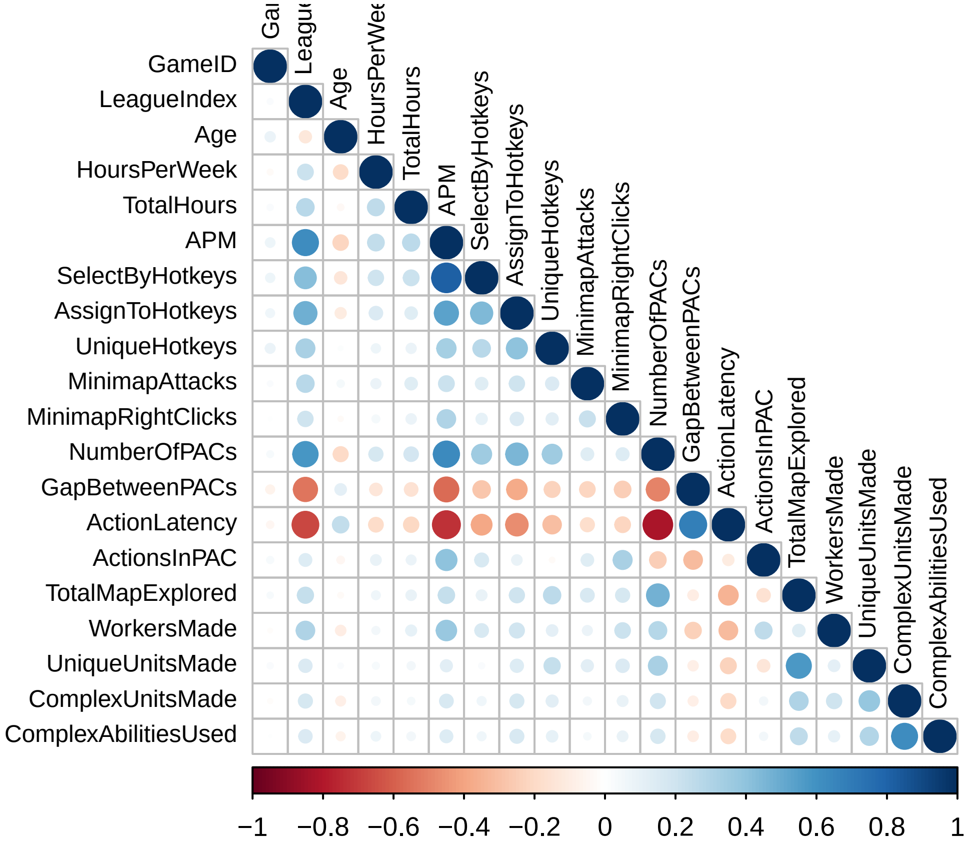
Correlation Plot

Using the correlation matrix provided below, we can see that LeagueIndex has a relatively strong correlation with **APM, SelectByHotkeys, AssignToHotkeys, NumberofPACs, GapBetweenPACs, and Action Latency**. Some of these predictors may be the best choices for the model. However, it's worth noting that many of the predictor values also have reasonably strong correlations within themselves, which may cause multiple colinearities in a model. This is not too surprising because many of these metrics capture the rate of actions in slightly different forms. For example, **APM** and **NumberOfPacs** likely have a strict mathematical relationship where approximately.

$$NumberOfPACs \approx APM * MatchDurationMinutes$$

The slight differences between these metrics could have some deep explanatory power, but that level of exploration is beyond the scope of this analysis. Focusing exclusively on **APM** fits into an Occam's razor approach by minimizing $span(X)$.

The following columns will be dropped as they may confound with **APMs, ActionLatency, GapBetweenPACs,**[1] **NumberofPACS, SelectbyHotkey, and ActionsInPAC.**



Visual Trend Analysis

Visually determining trends between the predictors and responding with an ordinal response is best done with alternatives to scatter plots. Violin plots will be used to gauge the linearity concerning the response and distribution with the variable at the varying levels ("A Complete Guide to Violin Plots | Tutorial by Chartio" n.d.). The appendix covers more details concerning why Violin plots were chosen.

MinimapAttacks, HoursPerweek, TotalHours, MinimapRightClicks, ComplexUnitsMade, ComplexAbilitiesUsed all have very long right tails. In search of gaussian predictors, these predictors could be transformed for linearity. Although a transform would have affected the explanation's simplicity.[2]

No Relationship **Age** the mean age of 22 does not vary much across **LeagueIndex** such that there is no stark linear relationship. However, the variance at the highest level seems to be much narrower than that at the lower levels.

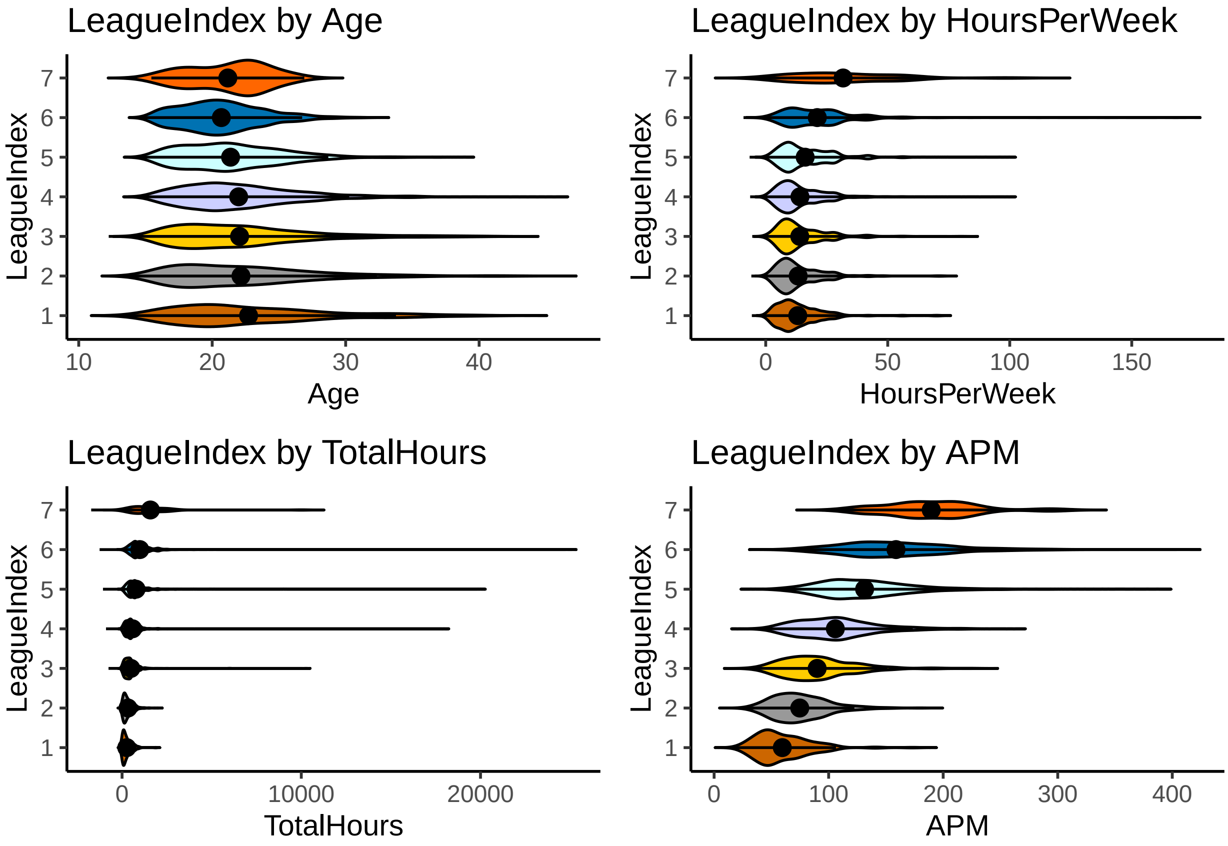
[1] An additional issue with this predictor is that it does not seem to line up with the time units in the description. Before and after the unit transformation **GapBetweenPACs** results in a mean is 11.3094444 hours.)

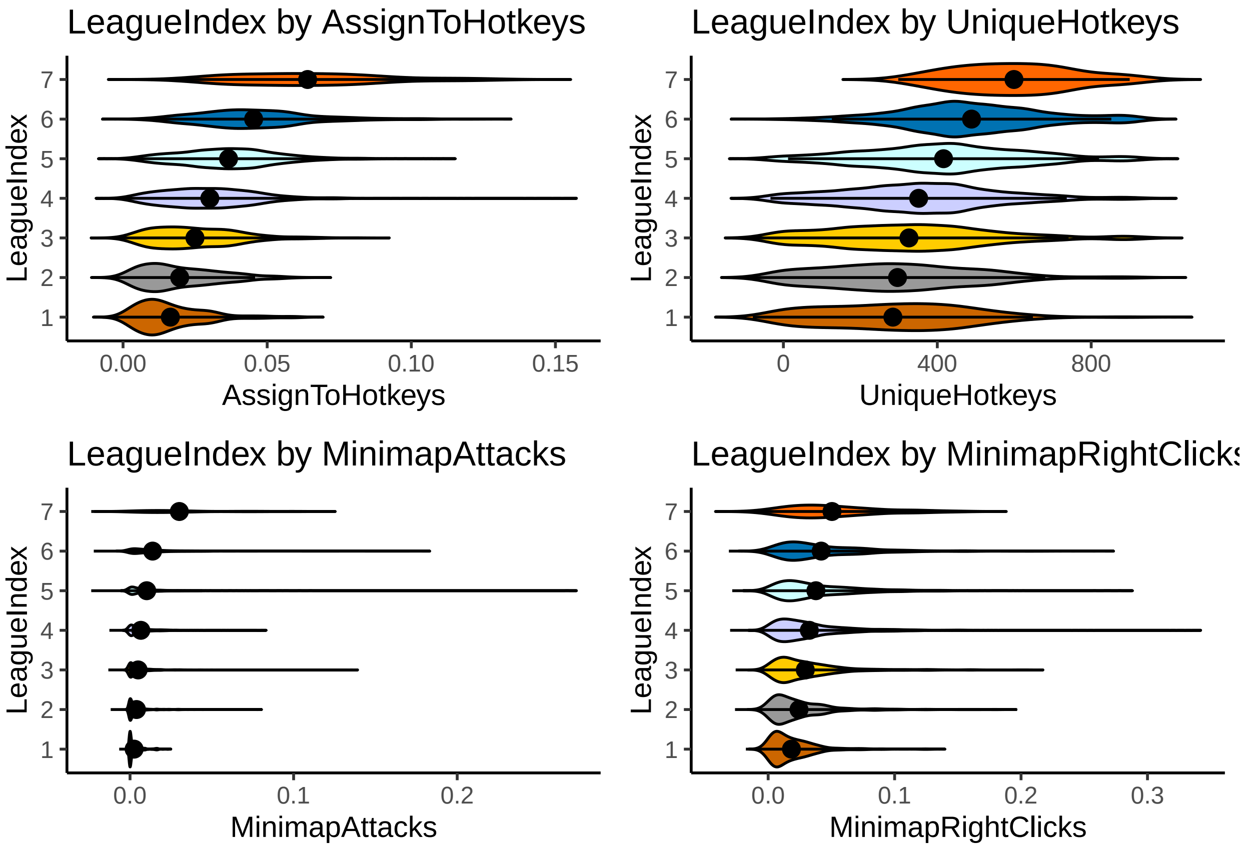
[2] A log transformation would be preferable, but enough observations by GameID that contain at least 0 in the related predictor entry would have to drop observations containing $-\infty$. If a transformation is pursued in the second half of this analysis, it will likely be a square root transformation

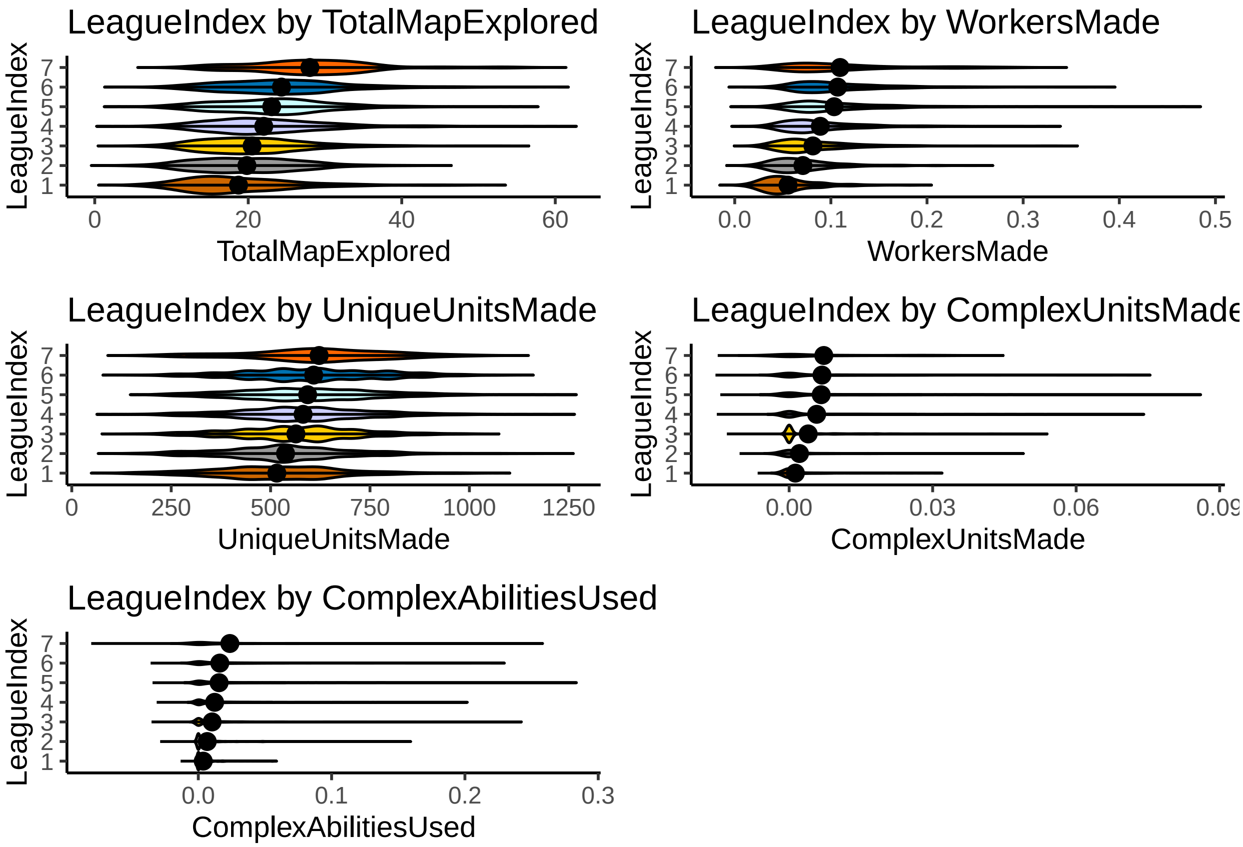
Bimodal **HourPerWeek** has visibly little or no relation to **LeagueIndex** 1-4, where **LeagueIndex** 4-7 seems to have a visible linear trend resulting in 0.03 vs 0.25. If **HoursPerWeek** survives the model trimming, it's bimodality may cause issues with the model's assumption $COV(Y) = \sigma^2 I$.**Workersmade, ComplexUnitsMades, and ComplexAbilityUsed** both have similar differences between **LeagueIndex** 1-4 and 4-7 with the portions that have no relation and a linear relation swapped in comparison to **HoursPerWeek**.

Linear **TotalHours, MinimapRightClicks, TotalMapExplored, and UniqueUnitsMade** have a positive linear trend with the response. **APM** has a strong linear relationship with the response.

Root **AssignToHotkeys, UniqueHotkeys, and MinimapAttacks** have a unique square root relationship with the response. This also may cause issues with the Gaussianity of the model's residuals.




LeagueIndex by AssignToHotkeys
LeagueIndex
7
6
5
4
3
2
1
0.00
0.05
0.10
0.15
AssignToHotkeys
LeagueIndex by UniqueHotkeys
LeagueIndex
7
6
5
4
3
2
1
0
400
800
UniqueHotkeys
LeagueIndex by MinimapAttacks
LeagueIndex
7
6
5
4
3
2
1
0.0
0.1
0.2
MinimapAttacks
LeagueIndex by MinimapRightClick
LeagueIndex
7
6
5
4
3
2
1
0.0
0.1
0.2
0.3
MinimapRightClicks



Model Specifications

Multivariant Regression Model Manual Model Iterations (Ω to ω)

A model with all predictors will be made. Subsequently, predictors will be dropped one by one until only predictors with significance of at least $\alpha = 5\%$ remain starting with lm_Ω and ending with lm_ω. The results are as follows:

The initial model has many insignificant predictors. The lm_Ω summary:

Table 1: Summary of Starting Manual Stepwise Backward Model Selection lm_Ω

term	estimate	std.error	statistic	p.value
(Intercept)	1.4609189	0.1366369	10.6919801	0.0000000
Age	-0.0022911	0.0045767	-0.5005919	0.6166916
HoursPerWeek	0.0051108	0.0016438	3.1091408	0.0018922
TotalHours	0.0001655	0.0000228	7.2461655	0.0000000
APM	0.0123241	0.0005256	23.4484478	0.0000000
AssignToHotkeys	13.4224183	1.2322640	10.8924862	0.0000000
UniqueHotkeys	0.0004731	0.0001013	4.6709082	0.0000031
MinimapAttacks	11.1976240	1.3853485	8.0828932	0.0000000
MinimapRightClicks	-0.7755611	0.6249251	-1.2410465	0.2146762
TotalMapExplored	0.0059030	0.0031347	1.8831294	0.0597701

term	estimate	std.error	statistic	p.value
WorkersMade	2.7112631	0.4407556	6.1513976	0.0000000
UniqueUnitsMade	0.0000345	0.0001434	0.2404174	0.8100215
ComplexUnitsMade	1.9538597	2.5075503	0.7791906	0.4359229
ComplexAbilitiesUsed	1.4388703	1.0068550	1.4290741	0.1530770

Age, UniqueUnitsMade, ComplexUnitsMade, MinimapRightClicks, and TotalMapExplored were removed in that order across the model's five iterations. All remaining predictors were significant to the predetermined α. The final iteration provided:

Table 2: Summary of Final Manual Stepwise Backward Model Selectionlm_{ω}

term	estimate	std.error	statistic	p.value
(Intercept)	1.5073472	0.0584345	25.795485	0.0000000
HoursPerWeek	0.0052759	0.0016259	3.244952	0.0011863
TotalHours	0.0001656	0.0000228	7.260133	0.0000000
APM	0.0123335	0.0005052	24.412493	0.0000000
AssignToHotkeys	13.5401278	1.2312698	10.996881	0.0000000
UniqueHotkeys	0.0005142	0.0000988	5.202741	0.0000002
MinimapAttacks	11.1888394	1.3551141	8.256751	0.0000000
WorkersMade	2.7573357	0.4331283	6.366094	0.0000000
ComplexAbilitiesUsed	2.3408788	0.7983207	2.932254	0.0033881

Assessing Fit and Overall Significance

A test will be performed to see if the predictors provide statistically significant better model than the null model $Y = \bar{Y} + \epsilon$ where $Y = LeagueIndex$. The hypothesis are as follows:

$$H_o : \beta = 0$$

$$H_a : LeagueIndex = X\beta + \epsilon$$

Without much surprise using 13 predictor variables results of a very small p-value of ~0. Over the iteration, this does not change notably across the other models as the last model also results in a p-value ~0. Thus all $\Delta p = p_{\Omega} - p_{\omega}$ iterations of the model, we can reject the null hypothesis suggesting that we should further investigate the explanatory power of our alternative hypothesis.

Testing for Significance Between Models

If both models had normal residuals, an F-test could be used on lm_{Ω} and lm_{ω} to determine if the models have significantly different residuals. RSS_{Ω} and RSS_{ω} both have Shapiro-Wilk's test statistics that reject the null at $\alpha = 5\%$ shown in a later section that examines the normality of each models' residuals. Without gaussian residuals conducting an ANOVA will be the only practice exercise for the final where the hypothesis is provided by:

$$H_o : RSS_{\omega} = RSS_{\Omega}$$

$$H_a : RSS_{\omega} \neq RSS_{\Omega}$$

Performing an ANOVA test below, we find an insignificant difference in the models at $\alpha = 5\%$ such that we cannot reject the H_o. The implications not rejecting H_o is that regardless of trimming the predictor space by Δp predictors, lm_ω is expected to produce comparable residuals with a 5% chance this is a result of the sampling. Additionally, their $adjR^2$ is barely different. Where $adjR^2_\Omega = 0.462$ and $adjR^2_\omega = 0.4614$. Further analysis will be conducted to see if the predictive and explanatory power of the models differs past the magnitude of their residuals.

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
3322	3748.29	NA	NA	NA	NA
3327	3757.68	-5	-9.39	1.66	0.14

Confidence Intervals

The confidence interval Table[3] show only a few subtleties between models coefficients confidence intervals.

In the starting model **Age, MinimapRightClicks, UniqueUnitsMade, ComplexUnitsMade, and ComplexAbilitiesUsed** are all not significant based on their p-value, and their confidence intervals straddle 0. Interestingly enough, **ComplexAbilitiesUsed** was initially above the alpha value for significance but made it to the final model. This could be because of the removal of a confounding variable. The magnitude of this shift is reflected in its delta value and delta_width.

For comparison, two summary statistics were added as follows:

Where:

$$delta = \frac{(UL_\omega + LL_\omega) - (UL_\Omega + LL_\Omega)}{(UL_\omega + LL_\omega)} = \frac{MeanCI_\omega - MeanCI_\Omega}{MeanCI_\omega}$$

$$delta_{width} = \frac{(UL_\omega - LL_\omega) - (UL_\Omega - LL_\Omega)}{(UL_\omega - LL_\omega)}$$

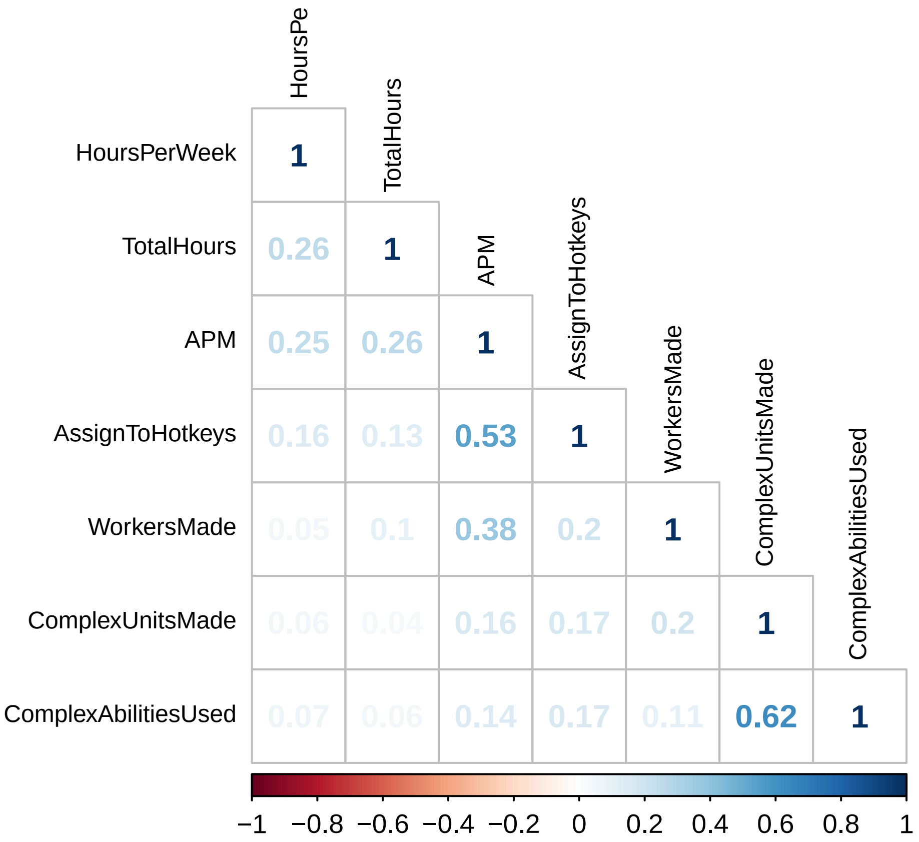
Table 4: Confidence Intervals Statistics at α 0.05

Row.names	LL_s	UL_s	LL_f	UL_f	mean_s	mean_f	delta	delta_width
(Intercept)	1.193	1.729	1.393	1.622	1.461	1.507	0.03	-1.34
Age	-0.011	0.007	NA	NA	-0.002	NA	NA	NA
APM	0.011	0.013	0.011	0.013	0.012	0.012	0.00	-0.04
AssignToHotkeys	11.006	15.838	11.126	15.954	13.422	13.540	0.01	0.00
ComplexAbilitiesUsed	-0.535	3.413	0.776	3.906	1.439	2.341	0.38	-0.26
ComplexUnitsMade	-2.963	6.870	NA	NA	1.954	NA	NA	NA
HoursPerWeek	0.002	0.008	0.002	0.008	0.005	0.005	0.03	-0.01
MinimapAttacks	8.481	13.914	8.532	13.846	11.198	11.189	0.00	-0.02
MinimapRightClicks	-2.001	0.450	NA	NA	-0.776	NA	NA	NA
TotalHours	0.000	0.000	0.000	0.000	0.000	0.000	0.00	0.00
TotalMapExplored	0.000	0.012	NA	NA	0.006	NA	NA	NA
UniqueHotkeys	0.000	0.001	0.000	0.001	0.000	0.001	0.08	-0.03
UniqueUnitsMade	0.000	0.000	NA	NA	0.000	NA	NA	NA
WorkersMade	1.847	3.575	1.908	3.607	2.711	2.757	0.02	-0.02

[3] Delta values are normalized by dividing by the mean of the model by related predictors confidence interval

Confounding

For reference in the following analysis of confounding variables, the new correlation matrix provides a zoomed view of the target of discussion.



Complex Units/Abilities As mentioned when exploring the confidence intervals, the notable change in **ComplexAbilitiesUsed**'s confidence interval between the lm_Ω to lm_ω is likely a result of the removal of the variable **ComplexUnitsMade**. Upon reintroducing **ComplexUnitsMade** into the final model, we find the following p-values for both predictors to be insignificant. The difference in P-value is a warning that we may not distinguish the effects or vary them independently. This aligns with what is expected based on the mechanics of the game. Players must make complex units before they can use their complex abilities.

The way forward with the model is to continue leaving out **ComplexUnitsMade** because only making a unit in SC2 is far from a win condition. Complex units must be utilized with precision and within the right contexts to reap their full value. On the other hand, worker units reflected in the predictor **WorkerUnitsMade** are units a player may produce and subsequently assign them to indefinite valued added work. If not interrupted by an attacking force, worker units will continue to add value to the in-game economy without additional intervention from the player as long as the resource node is still abundant.

Furthermore, using complex abilities is generally much more critical than making these complex units in masses. This also compliments the initial goal of the modeling to add flavor to APMs in a way that may reveal what actions are essential and fortunately APM and this **ComplexUnitsUsed** these there are mostly orthogonal with cor 0.14.

Table 5: ANOVA lm_ω and $lm_\omega + ComplexUnitsMade$

term	estimate	std.error	statistic	p.value
ComplexAbilitiesUsed	1.560207	1.003410	1.554905	0.1200637
ComplexUnitsMade	3.108488	2.420861	1.284043	0.1992165

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
3327	3757.68	NA	NA	NA	NA
3326	3755.82	1	1.86	1.65	0.2

AssigntToHotkeys and Actions Per Minute **AssignToHotkeys and APM** have the highest correlation within the final model. **AssignToHotkeys** was considered to be dropped along with the PAC related predictors prior. Still, I imagined the predictor added a significant flavor to what type of actions regardless of its potential for confounding.**AssignToHotkeys** in-game is when a player assigns units or buildings to hotkeys. For example, if the player has two armies within a match, they may select all of the units in army one and use the hotkey combination *CTRL+1* to assign those units to hotkey *1* for future rapid selection. Then the same player can select their second army use *Ctrl+2* to hotkey *2*. This works for any commandable unit or building in-game, allowing the player to reshape hotkeys based as assets are gained or lost.

Both variables will be removed from the model one at a time and then compared to the final model that contains both. In both subset models' **RSS** are statistically significant different. In terms of the impact to $adjR^2$, as **APM** seems to a explain a significant amount more than **AssignToHotkeys** as the models have with $adjR^2$ of 0.37 and 0.44 compared to the final model 0.46. To reaffirm the initial effort of dropping predictors that confound heavily with **APM** , **AssignToHotkeys** will be dropped from future modeling.

Table 7: ANOVA lm_ω and $lm_\omega - APM$

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
3327	3757.678	NA	NA	NA	NA
3328	4430.796	-1	-673.118	595.97	0

Table 8: ANOVA lm_ω and $lm_\omega - AssignToHotkeys$

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
3327	3757.678	NA	NA	NA	NA
3328	3894.264	-1	-136.586	120.931	0

Hours Metrics Upon closer examination of the remaining predictors its surprising that **TotalHours** and **HoursPerWeek** do not have a higher correlation. I did not expect both two make it to the final model. This may have to do with the reporting method. Regardless of the low correlation it is unclear if not impossible how to vary these two factors independently.

Both variables will be removed from the model one at a time and then compared to the final model that contains both. In both subset models' **RSS** are statistically significant different. In terms of the impact to $adjR^2$, as **TotalHours** seem to a explain a significant amount more than **HoursPerWeek** as the models have with $adjR^2$ of 0.45 and 0.46 compared to the final model 0.46. **HoursPerWeek** will be dropped for the final model.

Table 9: ANOVA lm_ω and $lm_\omega - TotalHours$

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
3327	3757.678	NA	NA	NA	NA
3328	3817.211	-1	-59.533	52.71	0

Table 10: ANOVA lm_ω and $lm_\omega - HoursPerWeek$

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
3327	3757.678	NA	NA	NA	NA
3328	3769.571	-1	-11.893	10.53	0.001

Revised lm_ω After performing the changes for confounding variables, the final model results in the following coefficients. This model will be rummaged through more expensively with better tools for the final portion of this assignment.

Table 11: lm_ω-HoursPerWeek-AssignToHotkeys

term	estimate	std.error	statistic	p.value
(Intercept)	1.6066666	0.0573980	27.991681	0.0e+00
TotalHours	0.0001764	0.0000228	7.743549	0.0e+00
APM	0.0148680	0.0004640	32.046258	0.0e+00
UniqueHotkeys	0.0008002	0.0000970	8.244981	0.0e+00
MinimapAttacks	12.5859243	1.3761559	9.145711	0.0e+00
WorkersMade	2.6528788	0.4410627	6.014744	0.0e+00
ComplexAbilitiesUsed	3.3072502	0.8095181	4.085456	4.5e-05

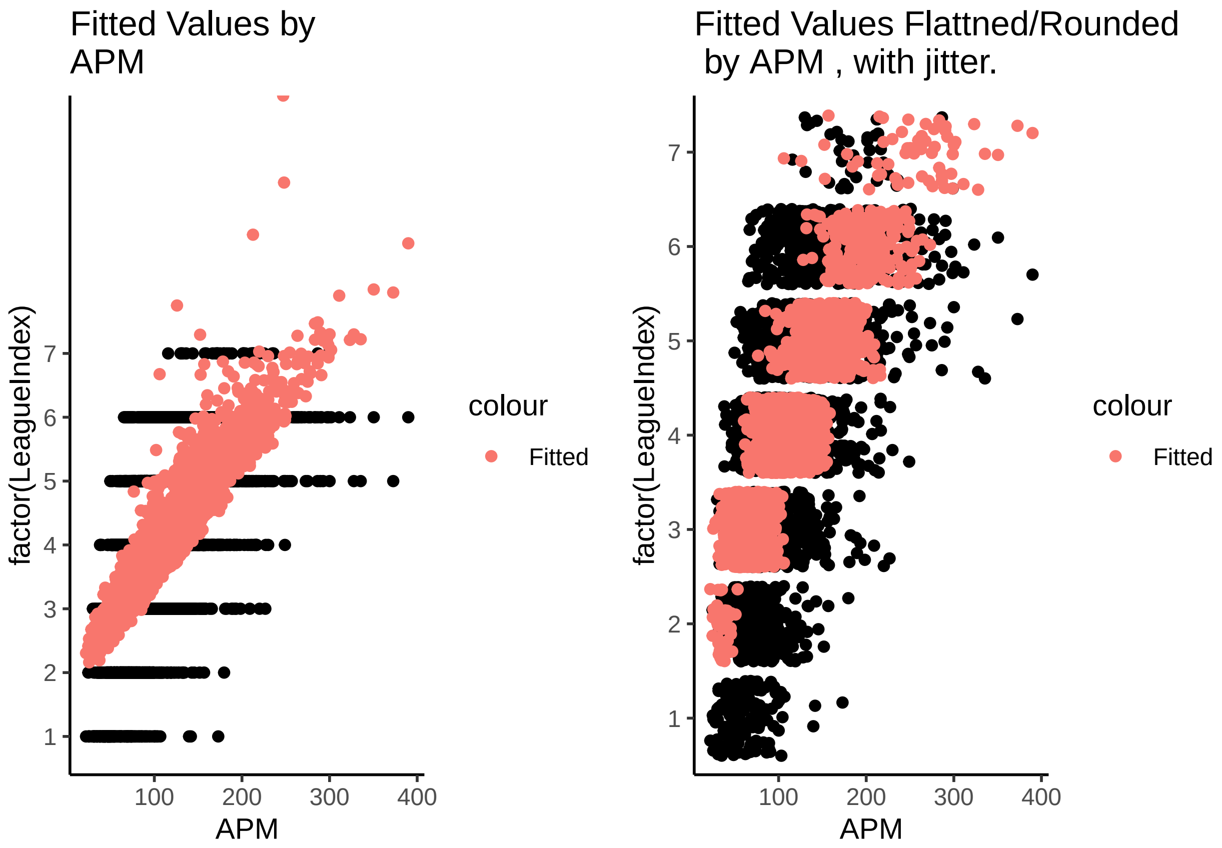
	2.5 %	97.5 %
(Intercept)	1.4941277	1.7192055
TotalHours	0.0001317	0.0002211
APM	0.0139584	0.0157777
UniqueHotkeys	0.0006099	0.0009904
MinimapAttacks	9.8877272	15.2841213
WorkersMade	1.7880975	3.5176601
ComplexAbilitiesUsed	1.7200469	4.8944536

Structural Uncertainty and Predictions

Without using cross-validation, we can see some fundamental issues with the fitted values of each model reviewed. The predictions of the model behavior and accuracy comparable between Ω & ω, so only ω will be used in the following section.

The first plot is a scatterplot with the fitted values by dominant predictor APM. Some of the players are excepted to have a LeagueIndex > 10, which does not exist in sampled response. Although LeagueIndex > 7 does tease that the Grandmaster tier is an unbounded tier in terms of skill points. Overall, the continuous fitted values fail to capture the discrete nature of the response.

The fitted values will be flattened such that when LeagueIndex > 7 will be set LeagueIndex=7, then each value will be rounded, so only integer values remain to help interpret the results of the model.

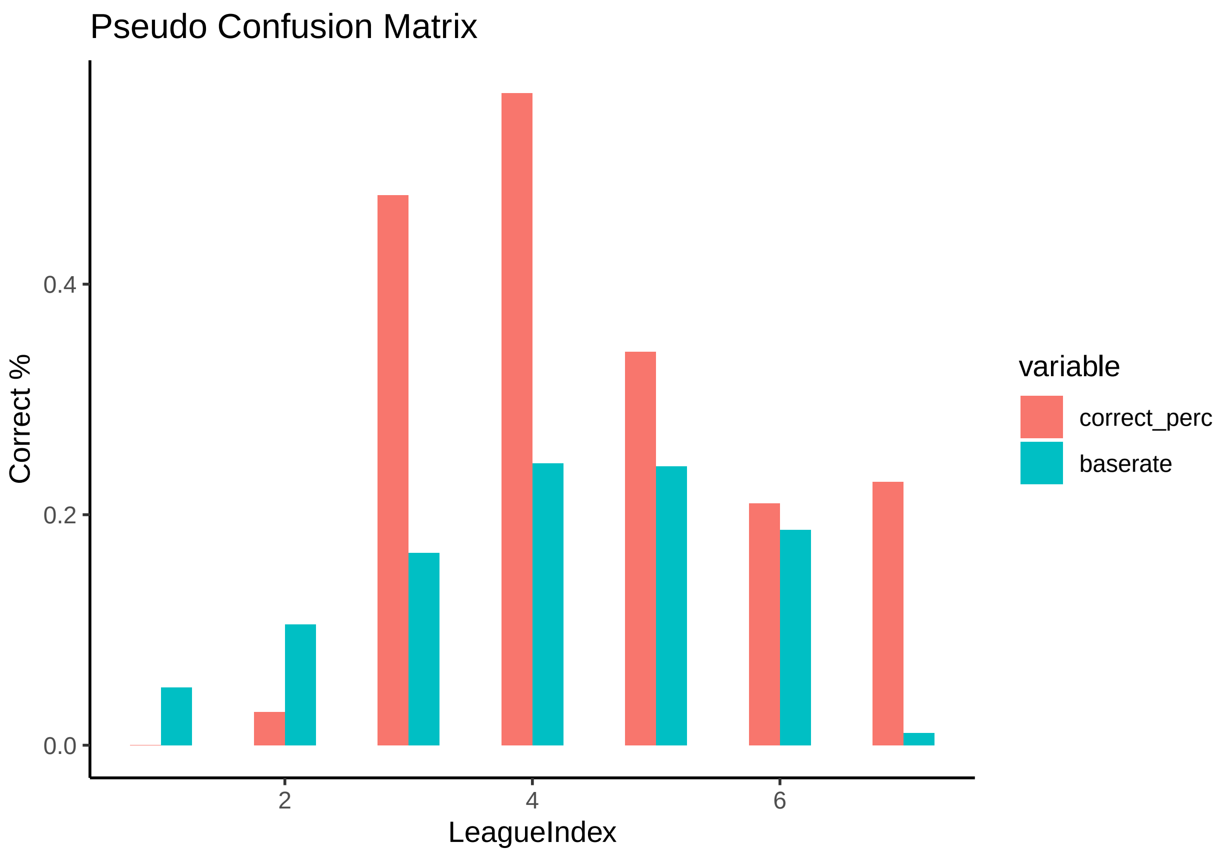


To assess the predictive accuracy of the model the flattened and rounded values fitted values are then compared to the sampled values overall providing a 34% accuracy. This is a very low accuracy considering 100% of the data was used to train the model. The correct % by **LeagueIndex** is plotted below accompanied by a base rate % that reflects the chance of guessing the correct **LeagueIndex** using:

$$BaseRate\% = n_{LeagueIndex=i}/n_{total}$$

.

```
## `summarise()` ungrouping output (override with `.groups` argument)
```



Another part of the models bias is that it does not place anyone below **LeagueIndex**= 2. The behavior of the ordinal response severely limits the predictive power of this model. The model has infinite more likelihood of predicting someone as **LeagueIndex** = 4 who is actually **LeagueIndex**=4 then predicting someone in **LeagueIndex**=1 who is actually **LeagueIndex**=1. Less severe lopsidedness can be seen through the varying **LeagueIndexs' Correct %**. Specifically, **Correct %** for **LeagueIndex** 3 to 7 perform much better than the base rate. These sign shows are the predictive power is heavily biased towards higher **LeagueIndex**

Conclusion

Research's Explanatory Power

The initial goal was to add some flavor to **APM** concerning what makes players highly ranked. Due to structure issues, this model overall performs very poorly. As a result, it needs to be kept in mind that the following predictors explain less than half the variation observed in the response.

From this analysis, we can see that the amount in terms of **TotalHours** a player dedicates seems to a significant predictor, which is a good sign for players willing to invest time into their craft. The suggested rate of progress is a bit disheartening as we expected the mean response of **LeagueIndex** to increase by 1 per 5000 **TotalHours**.

Using the predictor **WorkersMade**, one worker every 2.65 seconds is expected to increase mean LeagueIndex by one. Workers drive a player's economy, and the professionals always seems to have them in enormous quantities.

One minimap attack per 12.6 seconds is expected to increase mean **LeagueIndex** by one. Minimap attacks save the player from changing their main view for each attack command by allowing them to command a unit to any point using the minimap. For example, suppose we used the hotkey assignments mentioned previously. In that case, a player could press *1* and attack-click somewhere on the minimap to command an entire army to attack a location without changing the player's field of view. In a match where there are skirmishes across the map, a player more skilled at minimap attacks would be expected to have a strong advantage in managing multiple battlefronts near simultaneously. This may be a good target skill for players trying to increase their **LeagueIndex**.

One complex ability used per 3.3 seconds is expected to increase mean **LeagueIndex** by one. Even without the confounding variable in this model, it is still difficult to translate this idea into in-game practice because these Complex units need to be made before their abilities can be used. To construct these used, a player needs to have a relatively strong economy that may not be obtainable until midmatch.

It would take 1250 unique hotkeys used per second to increase mean **LeagueIndex** by one. This variable's units are incorrect or the effect is feeble. No additional information was provided by the data source to further troubleshoot ("UCI Machine Learning Repository: SkillCraft1 Master Table Dataset Data Set" n.d.).

Finally, **APM** increases **LeagueIndex** by one per 68. It's not surprising that speed seems to have a dominating effect.

Prelude to final

For the final, the logistical regression will remodel the same problem with a different set of techniques and assumptions that fit the ordinal response. The second iteration is likely to be much smoother because of this exploratory analysis's heavy leg work.

– End Midterm 2 –

Transition to the Final

There are five things that will be completed to fulfill the final portion of this paper, they are as follows:

Portional Odds model

Construct a ordinal regression model with the same predictors as final model lm_ω of the form:

$$log(\frac{p}{1-p}) = \alpha + \beta_1 x_1 + \beta_2 x_2 ... + \beta_p x_p vs. log(\frac{p_i}{1-p_i}) = \alpha_i + \beta_1 x_1 + \beta_2 x_2 ... + \beta_p x_p$$

Logistic regression predicts p,the probability a binomial response, alternatively ordinal regression predicts p_i the predictors ceoffients partitioned by multinominal response $i \in LeagueIndex$. This encoding predicts the logit odds based on a set of parameters X.

```r
require(foreign)
```

```
## Loading required package: foreign
```

```
## Warning: package 'foreign' was built under R version 4.0.3
```

```r
require(ggplot2)
require(MASS)
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
require(Hmisc)
require(reshape2)

m <- polr(factor(LeagueIndex) ~ APM + TotalHours, data = sc, Hess=TRUE)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
ctable <- coef(summary(m))
## calculate and store p values
p <- pnorm(abs(ctable[, "t value"]), lower.tail = FALSE) * 2
## combined table
ctable <- cbind(ctable, "p value" = p)

ci <- exp(confint(m)) # default method gives profiled CIs
```

```
## Waiting for profiling to be done...
```

```
co <-exp(coef(m))

kable(ctable,format = "markdown", digits = 4, caption="Ordinal Model" )
```

Table 13: Ordinal Model

	Value	Std. Error	t value	p value
APM	0.0307	0.0009	33.7002	0.0000
TotalHours	0.0008	0.0001	12.1164	0.0000
1|2	0.1529	0.1097	1.3940	0.1633
2|3	1.6048	0.0956	16.7790	0.0000
3|4	2.8441	0.0995	28.5892	0.0000
4|5	4.2950	0.1127	38.1056	0.0000
5|6	6.0066	0.1342	44.7461	0.0000
6|7	10.1836	0.2626	38.7769	0.0000

```
kable(co,caption="Exponentiate Coeffiecients")
```

Table 14: Exponentiate Coeffiecients

	x
APM	1.031158
TotalHours	1.000753

```r
kable(ci,caption="Exponentiate Confidence Intervals")
```

Table 15: Exponentiate Confidence Intervals

	2.5 %	97.5 %
APM	1.029329	1.033017
TotalHours	1.000613	1.000896

```r
#https://stats.idre.ucla.edu/r/dae/ordinal-logistic-regression/
#The intercept term shows cumulative odds
#melt, group by , summarize
```

To generate bias P-Values a z-test is used where the t-value is compared with the standard normal distribution. These p-values have decreasing bias as sample size increases and considering $n=$3336 the bias should be reasonably diminished producing subtlety asymmetric confidence intervals.

APM

For Starcraft II players this ranked season, the odds of being ranked above

MLR vs Proportional Odds Model (shallow dive)

- Briefly Compare the previous model and new models power using the same predictors, because the following models analyzed may not be a subset of one another.
- Revisit model specifications with a step wise reg focused on BIC
- Perform diagnostics
  - Adjust for outliers, and iterate
  - Perform diagnostics for portional odds, and iterate
- Predict

** OR **

- Perform diagnostics
  - Adjust for outliers, and iterate
  - Perform diagnostics on the residuals, and iterate
- Construct a initial ordinal regression model with the same predictors as lm_ω.
- Briefly Compare the previous model and new models power using the same predictors, because the following models analyzed may not be a subset of one another.
- Revisit model specifications with a step wise reg focused on BIC
- Assess porional odds assumptions
  - "One of the assumptions underlying ordinal logistic (and ordinal probit) regression is that the relationship between each pair of outcome groups is the same. In other words, ordinal logistic regression assumes that the coefficients that describe the relationship between, say, the lowest versus all higher categories of the response variable are the same as those that describe the relationship between the next lowest category and all higher categories, etc. This is called the proportional odds assumption or the parallel regression assumption. Because the relationship between all pairs of groups is the same, there is only one set of coefficients."

Appendix

About Columns

Attribute Information ("UCI Machine Learning Repository: SkillCraft1 Master Table Dataset Data Set" n.d.) :

1. GameID: Unique ID number for each game (integer)
2. LeagueIndex: Bronze, Silver, Gold, Platinum, Diamond, Master, GrandMaster, and Professional leagues coded 1-8 (Ordinal)
3. Age: Age of each player (integer)
4. HoursPerWeek: Reported hours spent playing per week (integer)
5. TotalHours: Reported total hours spent playing (integer)
6. APM: Action per minute (continuous)
7. SelectByHotkeys: Number of unit or building selections made using hotkeys per timestamp (continuous)
8. AssignToHotkeys: Number of units or buildings assigned to hotkeys per timestamp (continuous)
9. UniqueHotkeys: Number of unique hotkeys used per timestamp (continuous)
10. MinimapAttacks: Number of attack actions on minimap per timestamp (continuous)
11. MinimapRightClicks: number of right-clicks on minimap per timestamp (continuous)
12. NumberOfPACs: Number of PACs per timestamp (continuous)
13. GapBetweenPACs: Mean duration in milliseconds between PACs (continuous)
14. ActionLatency: Mean latency from the onset of a PACs to their first action in milliseconds (continuous)
15. ActionsInPAC: Mean number of actions within each PAC (continuous)
16. TotalMapExplored: The number of 24x24 game coordinate grids viewed by the player per timestamp (continuous)
17. WorkersMade: Number of SCVs, drones, and probes trained per timestamp (continuous)
18. UniqueUnitsMade: Unique unites made per timestamp (continuous)
19. ComplexUnitsMade: Number of ghosts, infestors, and high templars trained per timestamp (continuous)
20. ComplexAbilitiesUsed: Abilities requiring specific targeting instructions used per timestamp (continuous)

Why Violin Plots.

I decided to use Violin plots because I found with less tweaking they provided almost all the information I was looking for compared to scatter plots. Head to head a limitation of violin plots is that they make it seems as though the LeagueIndex level size contains the same n. The histogram earlier in this analysis shows clearly that n at each level of the **LeagueIndex** is not equal so choosing a tool on the basis of reiterating that point seems redundant. The benefit of Violin plots is that they provide a smoothed density plot at each **LeagueIndex** with a single point that represents the mean. This same thing could be done with scatter plots but I found it took much more staring and plot to plot variation.

The following is some head to head varieties plotting the data.

Code

```r
library(tidyverse)
library(dbplyr) #piping
library(ggplot2) #plotting
library(gridExtra)# easy plot grids
library(Hmisc) # for correlation matrix
library(corrplot) # For correlation matrix graphic
library(broom) #tidy lm summaries
library(knitr) #pretty tables
library(reshape2) #melt function

sc<-read_csv("~/STAT757/skillcraft/SkillCraft1_Dataset.csv")

colnames(sc)

cbPalette <- c("#CC6600", "#999999", "#FFCC00", "#CCCFFF", "#CCFFFF","#0072B2", "#FF6600")

#set type
sc$HoursPerWeek<-as.numeric(sc$HoursPerWeek)
sc$TotalHours<-as.numeric(sc$TotalHours)

count_missing_age<-count(sc%>%
  filter(is.na(Age))%>%arrange(LeagueIndex))
count_professional<-count(sc%>%filter(LeagueIndex==8))
count_grandmaster<-count(sc%>%filter(LeagueIndex==8))
print(paste(’There are ’,count_missing_age,’ missing values in the age column. There are ’,count_profess

sc<-filter(sc,sc$TotalHours<1000000)

sc<-sc%>%
  drop_na()%>%
  filter(HoursPerWeek!=0)
sc_describe<-describe(sc)

sc<-sc%>%
  mutate(NumberOfPACs=NumberOfPACs*88.5,
         MinimapAttacks=MinimapAttacks*88.5,
         MinimapRightClicks=MinimapRightClicks*88.5,
         SelectByHotkeys=SelectByHotkeys*88.5,
         AssignToHotkeys=AssignToHotkeys*88.5,
         UniqueHotkeys=UniqueHotkeys*88.5,
         WorkersMade=WorkersMade*88.5,
         UniqueUnitsMade=UniqueUnitsMade*88.5,
         ComplexUnitsMade=ComplexUnitsMade*88.5,
         ComplexAbilitiesUsed=ComplexAbilitiesUsed*88.5,
         GapBetweenPACs=GapBetweenPACs*1000,
         ActionLatency=ActionLatency*1000)


LeagueIndex_Normal<-shapiro.test(sc$LeagueIndex)

ggplot(sc)+
  geom_histogram(aes(x=LeagueIndex,y=(..count..)/sum(..count..),fill=LeagueIndex),
      position = "identity", binwidth = 1,fill=cbPalette) +
  ylab("Relative Frequency")+
  ggtitle(’LeagueIndex Distribution’,subtitle = paste(LeagueIndex_Normal[3],
      " P-Value: ",LeagueIndex_Normal[2]))+xlab("LeagueIndex 1-Bronze to 7-Grandmaster")+theme_classic(
sc_cor<-cor(select_if(sc,is.numeric),use = "complete.obs")
sc_cor_plot<-corrplot(sc_cor,
    tl.cex=.75,
    tl.col=’black’,
    type="lower",)

sc<-sc%>%select(!c(GameID,ActionLatency,GapBetweenPACs,NumberOfPACs,SelectByHotkeys,ActionsInPAC))

cor_hoursperweek<-paste(
  round(cor(sc%>%filter(between(LeagueIndex,1,4))%>%select(LeagueIndex),
    sc%>%filter(between(LeagueIndex,1,4))%>%select(HoursPerWeek))[1],
    2),
  "vs",
  round(cor(sc%>%filter(between(LeagueIndex,4,7))%>%select(LeagueIndex),
    sc%>%filter(between(LeagueIndex,4,7))%>%select(HoursPerWeek))[1],
    2)
)

VioLeagueIndex<-function(predictor){
  ggplot(sc, aes(x=factor(LeagueIndex), y=unlist(sc%>%select(all_of(predictor))), fill=factor(LeagueInde
    geom_violin(trim=FALSE, color="black")+scale_fill_manual(values=cbPalette)+
    stat_summary(fun.data=mean_sdl,geom="pointrange", color="black")+ coord_flip()+
    ggtitle(paste("LeagueIndex by",predictor))+
    xlab("LeagueIndex")+ylab(predictor) + guides(fill= FALSE)+theme_classic()
    }

plots<-lapply(colnames(sc)[2:length(colnames(sc))],VioLeagueIndex)

do.call("grid.arrange", c(plots[1:4], ncol=2))
do.call("grid.arrange", c(plots[5:8], ncol=2))
do.call("grid.arrange", c(plots[9:13], ncol=2))

sc_lm<-lm(LeagueIndex~.,sc)
sc_lm_1<-update(sc_lm,.~.-UniqueUnitsMade ,sc)
sc_lm_2<-update(sc_lm,.~.-Age-UniqueUnitsMade)
sc_lm_3<-update(sc_lm,.~.-Age-UniqueUnitsMade-ComplexUnitsMade)
sc_lm_4<-update(sc_lm,.~.-Age-UniqueUnitsMade-ComplexUnitsMade-MinimapRightClicks)
sc_lm_final<-update(sc_lm,.~.-Age-TotalMapExplored-UniqueUnitsMade-MinimapRightClicks-ComplexUnitsMade)

kable(anova(sc_lm,sc_lm_final),digits=2)

#t<-data.frame(confint(sc_lm),confint(sc_lm_final))
t1 <- data.frame(confint(sc_lm))
t2 <- data.frame(confint(sc_lm_final))
t3<-merge(t1,t2,by="row.names",all=TRUE)
t3<-t3%>%rename(LL_s=X2.5...x,UL_s=X97.5...x,LL_f=X2.5...y,UL_f=X97.5...y)
t3<-t3%>%mutate(mean_s=(UL_s+LL_s)/2,
                mean_f=(UL_f+LL_f)/2,
                delta=(mean_f-mean_s)/mean_f,
                delta_width=((UL_f-LL_f)-(UL_s-LL_s))/(UL_f-LL_f))%>%
         mutate_if(is.numeric, ~round(., 3))
kable(t3,format = "markdown", digits = c(4,4,4,4,4,4,4,2,2), caption="Confidence Intervals Statistics a
sc_omega<-sc%>%select(HoursPerWeek,TotalHours,HoursPerWeek,APM,AssignToHotkeys,WorkersMade,ComplexUnits

sc_omega_cor<-cor(select_if(sc_omega,is.numeric),use = "complete.obs")
sc_omega_cor_plot<-corrplot(sc_omega_cor,
    tl.cex=.75,
    tl.col='black',
    method="number",
    type="lower")

sc_lm_5<-update(sc_lm_final,.~.+ComplexUnitsMade)
kable(tidy(sc_lm_5,conf.level = .05)[9:10,],caption = "ANOVA $lm_\\omega$ and $lm_\\omega+ComplexUnitsM
kable(anova(sc_lm_final,sc_lm_5),digits=2)

sc_lm_6<-update(sc_lm_final,.~.-APM)
sc_lm_7<-update(sc_lm_final,.~.-AssignToHotkeys)

kable(anova(sc_lm_final,sc_lm_6),digits=3,caption = "ANOVA $lm_\\omega$ and $lm_\\omega-APM$")
kable(anova(sc_lm_final,sc_lm_7),digits=3,caption = "ANOVA $lm_\\omega$ and $lm_\\omega-AssignToHotkeys

sc_lm_8<-update(sc_lm_final,.~.-TotalHours)
sc_lm_9<-update(sc_lm_final,.~.-HoursPerWeek)

kable(anova(sc_lm_final,sc_lm_8),digits=3,caption = "ANOVA $lm_\\omega$ and $lm_\\omega-TotalHours$")
kable(anova(sc_lm_final,sc_lm_9),digits=3,caption = "ANOVA $lm_\\omega$ and $lm_\\omega-HoursPerWeek$")

sc_lm_final<-update(sc_lm_final,.~.-AssignToHotkeys-HoursPerWeek)
kable(tidy(sc_lm_final,conf.level = .05),caption = "$lm_\\omega$-HoursPerWeek-AssignToHotkeys")
kable(confint(sc_lm_final))

sc_lm_rounded<-sc_lm_final
sc_lm_rounded$fitted.values[sc_lm_rounded$fitted.values>7]<-7
sc_lm_rounded$fitted.values<-round(sc_lm_rounded$fitted.values)

p1<-ggplot(sc_lm_final, aes(x=APM, y=factor(LeagueIndex)))+geom_point()+geom_point(aes(y=fitted(sc_lm_fi

p2<-ggplot(sc_lm_final, aes(x=APM, y=factor(LeagueIndex)))+geom_jitter() +geom_jitter(aes(y=round(sc_lm

grid.arrange(p1,p2,ncol=2)

sc_predicted<-data.frame("LeagueIndex"=sc$LeagueIndex,"FlattenFitted"=sc_lm_rounded$fitted.values)

sc_predicted$correct<-sc_predicted$LeagueIndex==sc_predicted$FlattenFitted

sc_predicted_agg<-sc_predicted%>%group_by(LeagueIndex)%>%
  add_tally()%>%summarise(correct_perc=sum(correct)/max(n),incorrect_perc=1-sum(correct)/max(n),n=max(n
  melt(id="LeagueIndex")

ggplot(sc_predicted_agg,aes(x=LeagueIndex,y=value,fill=variable))+
   geom_bar(stat="identity", width=.5, position = "dodge")+theme_classic()+ylab("Correct %")+ggtitle("P
```

References

"A Complete Guide to Violin Plots | Tutorial by Chartio." n.d. Accessed November 2, 2020. https://chartio.com/learn/charts/violin-plot-complete-guide/.

"AlphaStar: Mastering the Real-Time Strategy Game Starcraft Ii | Deepmind." n.d. Accessed November 2, 2020. https://deepmind.com/blog/article/alphastar-mastering-real-time-strategy-game-starcraft-ii.

"APM Definition." n.d. Accessed November 2, 2020. https://techterms.com/definition/apm.

"Congrats to the Starcraft Ii Wcs Global Finals Champion! - Blizzcon." n.d. Accessed October 26, 2020. https://blizzcon.com/en-us/news/23198508/congrats-to-the-starcraft-ii-wcs-global-finals-champion.

"ELO Rating Sysem in Chess." n.d. Accessed November 8, 2020. https://chance.amstat.org/2020/09/chess/.

"Leagues: 2019 - Liquipedia - the Starcraft Ii Encyclopedia." n.d. Accessed October 26, 2020. https://liquipedia.net/starcraft2/Battle.net_Leagues.

"Ordinal Logistic Regression | R Data Analysis Examples." n.d. Accessed October 27, 2020. https://stats.idre.ucla.edu/r/dae/ordinal-logistic-regression/.

"Perception-Action Cycle - Models, Architectures, and Hardware | Vassilis Cutsuridis | Springer." n.d. Accessed October 29, 2020. https://www.springer.com/gp/book/9781441914514.

"UCI Machine Learning Repository: SkillCraft1 Master Table Dataset Data Set." n.d. Accessed October 26, 2020. https://archive.ics.uci.edu/ml/datasets/SkillCraft1+Master+Table+Dataset.

"What Is a Smurf Account? Everything You Need to Know | Lol-Smurfs." n.d. Accessed November 2, 2020. https://www.lol-smurfs.com/blog/what-is-a-smurf-account.

"Winnings: 2019 - Liquipedia - the Starcraft Ii Encyclopedia." n.d. Accessed October 26, 2020. https://liquipedia.net/starcraft2/Winnings/2019.