STA 4320 Written Report
Starcraft 2 Analysis

Andres Gonzalez
Michael Tsui
Ariel De Leon

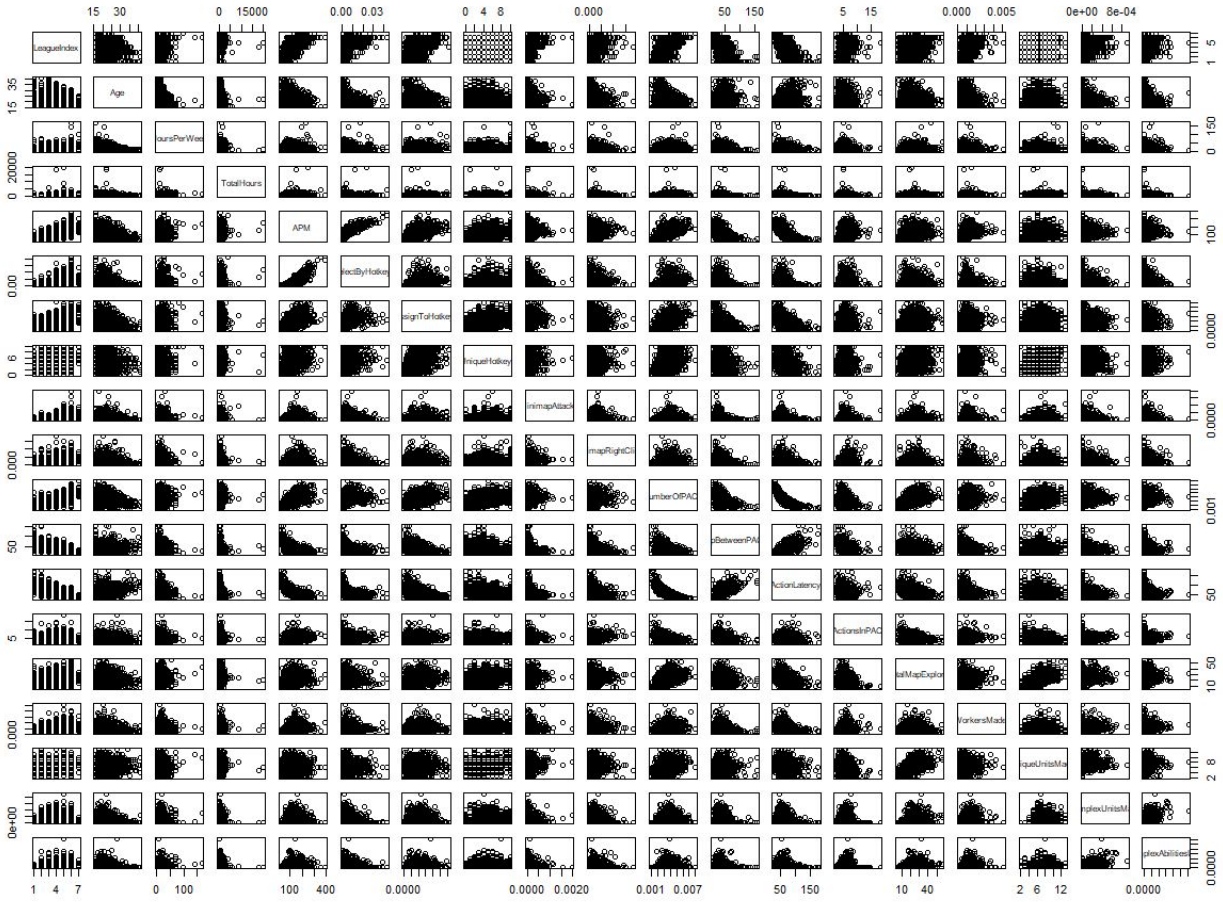
1. Introduction

Our goal for this project is to select the best model to predict League Index from the Starcraft2 data set. The response variable LeagueIndex is an ordinal variable described as Bronze, Silver, Gold, Platinum, Diamond, Master, GrandMaster, and Professional leagues coded 1-7; in addition, it is a ranking system based on the players level on the game. Now, our goal is to create the best predictive model to predict the overall skill of the player base with 18 various predictors which in this case we have about 18 predictors to fit into our model, and more importantly we want to analyze the data by using different methods and techniques to help us pick what will be the best model, and ultimately to provide the best explanation on what player attributes can be used as predictors to determine the League Index ranking system.

2. Data Description and Ideas

The data set of Starcraft2 has 3338 observations of 19 variables. Since there are a large number of observations we decided to split the observations (randomly) into two parts which are the training set and the validation set, and these observations are evenly distributed by 50 % for both sets. Each set contains 1669 observations of 19 variables. To get a quick glance at all of this, we plotted the full regular Starcraft data set to have an idea what predictors are going to be included into fitting a best fit model by just observing the plot, but it was hard to read and interpret which did not really help us. In addition, we want to select the best model to predict LeagueIndex using these 18 predictors by using the training data set, and proceed to a method for model selection that can potentially be used, such as LASSO, Ridge, and Best Subset Selection and so forth. From there we can do LOOCV, AIC, etc., on the training set to perform model selection. We choose to go with different methods for model selection to use on the training data using the appropriate linear regression techniques. Now, it will be very tedious to look at each plot based on all 18 predictors we are given, so starting with the full model is

the best way to analyze the data to start performing the modeling process, and compare the AIC on both the forward model selection and backward model selection to see which one has the lowest AIC, and because the number of predictors was quite large, we decided to use ridge and lasso for the other two models. Theoretically, Ridge works well if there are many large parameters around the same value while lasso works well if there are a small number of significant parameters but in practice since we do not know the true parameter value we will run a cross validation to select the better suited model.



3. Model Selection

We start performing the modeling process selection using the full training set model calling it m0, and try to use different methods for model selection. In this

case, the most popular methods are forward and backward selection procedures. Alternatively, the best subset selection is also a very popular method algorithm to use, but we decided to exclude it since there are 18 predictors (p=18), so the computational complexity will become too large. In that aspect, we could not do best subset selection on LeagueIndex prediction for the Starcraft training set because the algorithm took too long, due to having a large number of predictors it makes the computational cost difficult to find the best model. The forward and backward selection are both different in what they do to produce a model. With that being said, it will often produce the same model as each other (and best subset), but this is not guaranteed. Since, the forward selection begins with m0, 0 predictors, the algorithm works by adding each predictor at a time, and stops until what predictor/s offers no improvement. Hence, extracting the info, a 15th predictor offers no improvement afterwards, so it is the best model on basis of AIC of 4692.616. As a result, we called this model m1. Now, for the backward selection it has the same algorithm process like the forward selection but this time begins with m0, all p predictors and try removing one by one, and stops until what predictor/s offers no improvement. Extracting the info, we get the full model with an AIC of 4697.676, however this is not the model we are looking for. As we try removing predictors one by one based on the output information we get that the full

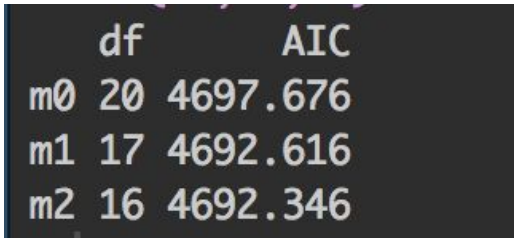
Selection Summary

Variable	AIC	Sum Sq	RSS	R-Sq	Adj. R-Sq
ActionLatency	5038.805	1623.493	1993.335	0.44887	0.44854
AssignToHotkeys	4889.088	1796.702	1820.126	0.49676	0.49616
APM	4819.001	1873.643	1743.185	0.51803	0.51717
MinimapAttacks	4768.671	1927.451	1689.377	0.53291	0.53179
TotalHours	4739.153	1959.054	1657.774	0.54165	0.54027
GapBetweenPACs	4721.649	1978.314	1638.514	0.54697	0.54534
UniqueHotkeys	4715.442	1986.352	1630.477	0.54920	0.54730
WorkersMade	4709.009	1994.569	1622.259	0.55147	0.54931
SelectByHotkeys	4704.675	2000.714	1616.114	0.55317	0.55074
HoursPerWeek	4701.894	2005.336	1611.492	0.55445	0.55176
Age	4698.943	2010.110	1606.718	0.55577	0.55282
NumberOfPACs	4697.599	2013.326	1603.502	0.55666	0.55344
UniqueUnitsMade	4694.904	2017.830	1598.998	0.55790	0.55443
ActionsInPAC	4692.735	2021.819	1595.009	0.55900	0.55527
ComplexAbilitiesUsed	4692.616	2023.843	1592.985	0.55956	0.55557

Backward Elimination Summary

Variable	AIC	RSS	Sum Sq	R-Sq	Adj. R-Sq
Full Model	4697.676	1592.088	2024.740	0.55981	0.55501
MinimapRightClicks	4695.690	1592.101	2024.727	0.55981	0.55527
ComplexUnitsMade	4693.981	1592.379	2024.449	0.55973	0.55547
TotalMapExplored	4692.616	1592.985	2023.843	0.55956	0.55557
APM	4692.346	1594.637	2022.191	0.55911	0.55537

model of predictors from MinimapRightClicks, ComplexUnitsMade, TotalMapExplored, and APM should not be included in the model in order to perform the best model on basis of AIC of 4692.346. In addition, we called this model m2. Finally, we can compare the AIC values for all models deciding on our model based on which one has the lowest value. Therefore, m2 is the best model on the basis of AIC of 4692.346.



	df	AIC
m0	20	4697.676
m1	17	4692.616
m2	16	4692.346

Generally, Lasso and Ridge are good methods when p is large. In this case, the training set contains 1669 observations and 18 predictors, so both methods are appropriate alternatives to any model selection procedure. The intention is to compare Ridge and Lasso models to the backwards selection model. When we dealt with the training data, we decided to compare the Leave-One-Out Cross Validation values for the four models (Ridge, Lasso, m1,m2) . The values for the Ridge, Lasso and backwards selection were 0.9872421, 0.9865503, and 0.9848855 respectively. At this point it appears the backwards selection is the best model on the basis of LOOCV.

	model	LOOCV
1	fit_ridge	0.9872421
2	fit_lasso	0.9865503
3	m1	0.9852695
4	m2	0.9848855

4. Testing Validation Data using MSE

So now we can finally test the performance of the three models by taking the estimates of Mean Squared Error for Ridge, Lasso and BS respectively. The results were 14.13684 for Ridge, 14.51549 for Lasso and 14.94856 for FS, 14.8740 for BS.

```
> c(TestMSE_ridge,TestMSE_lasso,TestMSE_FS,TestMSE_BS
[1] 14.13684 14.51549 14.94856 14.87407
```

Since Ridge produced the best MSE, we can now examine the coefficients and determine which predictors had the biggest influence.

5. Conclusion

Seeing that Ridge had the lowest MSE in terms of predicting the validation set, we can use a coefficient command to extract the predictors that did the best in predicting LeagueIndex.

```
19 x 2 sparse Matrix of class "dgCMatrix"
                              s0            s0
(Intercept)          3.937343e+00  4.316008e+00
Age                  1.272861e-02  1.402501e-02
HoursPerWeek         4.917299e-03  4.824116e-03
TotalHours           1.471368e-04  1.492306e-04
APM                  1.763116e-03  .
SelectByHotkeys      2.350255e+01  3.261235e+01
AssignToHotkeys      8.973900e+02  9.314444e+02
UniqueHotkeys        3.190804e-02  3.063447e-02
MinimapAttacks       1.073928e+03  1.111379e+03
MinimapRightClicks  -8.560984e+00  .
NumberOfPACs         1.781425e+02  1.642500e+02
GapBetweenPACs      -1.066608e-02 -9.816857e-03
ActionLatency       -2.162454e-02 -2.619428e-02
ActionsInPAC         2.139089e-02  3.186838e-02
TotalMapExplored    -2.465998e-03 -2.889087e-03
WorkersMade          1.695678e+02  1.762127e+02
UniqueUnitsMade     -2.898260e-02 -2.848983e-02
ComplexUnitsMade     1.431399e+02  1.107233e+02
ComplexAbilitiesUsed 1.430201e+02  1.249549e+02
```

From the output we see that MinimapRightClicks, GapBetweenPACs, ActionLatency, TotalMapExplored, and UniqueUnitsMade have a negative correlation with the response impling that these variables do nothing to improve the model. MinimapRightClicks and TotalMapExplored agrees with our early assumptions from the BS model. Likewise, the remaining positive coefficients indicate that there is some positive correlation between the predictors and the response and it should be noted though that the goal of the analysis is measuring overall skills using predictors.

Note: we did try adding in all Quadratic terms while trying to improve the forward selection model after doing it once already, but while the LOOCV improved and the curve for residuals vs fitted flattened. When we took the MSE value we got a huge number of 663k. So we decided to just use the original models given by the forward and backward selection.