Final Project Report: MGSC 310

Ben Ziv, Brad Shimabuku, Lucas Sotto, Anne Schmidt

MGSC 310-01: Statistical Models in Business Analytics

Professor Shahryar Doosti

May 22, 2021

Problem/Motivation

As professional sports teams are constantly looking to gain a competitive advantage over their opponents, the use of data analytics in sports is becoming a popular way for them to do so. In the game of baseball, statistics are a great way to measure player performance and by applying data science models, we can figure out the best ways to enhance these stats. This project was motivated by the ability of these models to be implemented for the benefit of real MLB hitters. The players can simply read the results of our models and apply them to their game, to improve their performance. Additionally, two group members play baseball for the University and can apply these models to their game as well.

Data Set

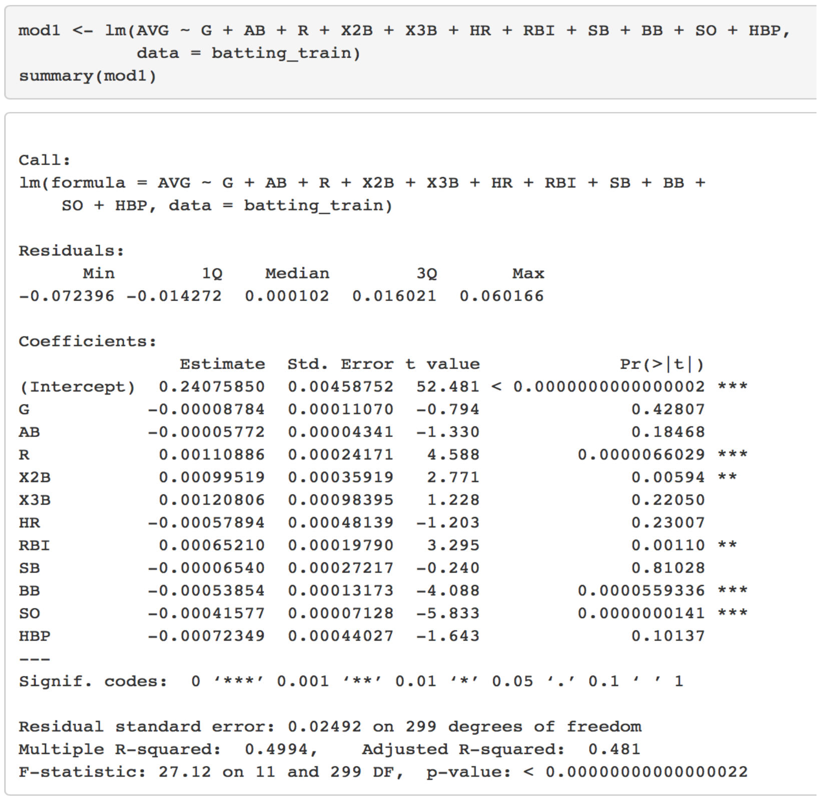
Our group used the Lahman Baseball Data Set from Kaggle to build analytical models to make predictions about different statistics. Within the dataset we used the batting statistics from the 2019 MLB season. Taking a glimpse of the data, there were 23 variables, mainly consisting of continuous variables with a few categorical variables as well. The variables we used in the models included games played (G), at bats (AB), runs (R), hits (H), doubles (X2B), triples (X3B), home runs (HR), runs batted in (RBI), stolen bases (SB), walks (BB), strikeouts (SO), hit by pitch (HBP), and batting average (AVG).

In order to clean our data to make it the most accurate for our motivating questions, we had to adjust a few specific attributes. We first had to filter the dataset to the 2019 MLB season because the original set provided statistics from many years in the past. Next we had to create (mutate) a new variable into the dataset, batting average (AVG). Batting average is the outcome variable we predicted for each model and was created by taking the number of hits divided by the number of at bats per player. A hitter's batting average gives a general idea of how well they're doing and is a stat that hitters are always looking to increase. A .300 AVG is universal for a "strong" hitter (MLB average in 2019 = .252 AVG). Finally, we filtered the number of at bats per player to be greater than 100 to make sure we wouldn't skew the data with players who may have a very high or low AVG, with a minimal number of at bats. To make predictions for batting average and answer motivating questions for practical use, we created three different models. These models included Linear Regression, Decision Trees, and Lasso Regression.

Model 1: Linear Regression

The goal of the linear regression model was to find out the relationships between the outcome and predictor variables and to determine which variables were significant in predicting AVG. A small summary stat to be aware of is hits vs. batting average. Logically, in the game of baseball, as a batter gets more hits, their batting average will increase. We wanted to express this direct relationship between these two variables because we will not include hits in the linear model predictions. This is because hits is the core of batting average and it could cause the model to overfit too much. We wanted to look at what other variables affect batting average.

Figure 1: Linear Regression Model Output



```
mod1 <- lm(AVG ~ G + AB + R + X2B + X3B + HR + RBI + SB + BB + SO + HBP,
           data = batting_train)
summary(mod1)
```

```

Call:
lm(formula = AVG ~ G + AB + R + X2B + X3B + HR + RBI + SB + BB +
    SO + HBP, data = batting_train)

Residuals:
      Min        1Q    Median        3Q       Max
-0.072396 -0.014272  0.000102  0.016021  0.060166

Coefficients:
               Estimate  Std. Error t value             Pr(>|t|)
(Intercept)  0.24075850  0.00458752  52.481 < 0.0000000000000002 ***
G           -0.00008784  0.00011070  -0.794              0.42807
AB          -0.00005772  0.00004341  -1.330              0.18468
R            0.00110886  0.00024171   4.588         0.0000066029 ***
X2B          0.00099519  0.00035919   2.771              0.00594 **
X3B          0.00120806  0.00098395   1.228              0.22050
HR          -0.00057894  0.00048139  -1.203              0.23007
RBI          0.00065210  0.00019790   3.295              0.00110 **
SB          -0.00006540  0.00027217  -0.240              0.81028
BB          -0.00053854  0.00013173  -4.088         0.0000559336 ***
SO          -0.00041577  0.00007128  -5.833         0.0000000141 ***
HBP         -0.00072349  0.00044027  -1.643              0.10137
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02492 on 299 degrees of freedom
Multiple R-squared:  0.4994,    Adjusted R-squared:  0.481
F-statistic: 27.12 on 11 and 299 DF,  p-value: < 0.00000000000000022
```

In the linear regression model, we chose to use G, AB, R, X2B, X3B, HR, RBI, SB, BB, SO, and HPB as the variables for predicting AVG. From the output of our model we can interpret how each variable is affected by our outcome variable batting average, based on their coefficient value and how statistically significant each one is based on their p-values. For example, we can interpret that, for every 1 unit increase in strikeouts (SO), there is a 0.000416 decrease in AVG, meaning that generally when a hitter is striking out often, their batting average will be lower. Another example we can interpret is that, for every 1 unit increase in runs (R), there is a 0.00111 increase in AVG, meaning that generally when a player is scoring more runs, they will tend to have a higher overall batting average. Looking at the p-values outputted from our model we can determine the significance of each variable. P-value is the probability of beta equaling zero. A small p-value indicates that there is an association/relationship between the x & y variables. The variables that are statistically significant are runs (R), doubles (X2B), runs batted in (RBI), walks (BB) and strikeouts (SO) because the typical p-value for rejecting the null is 5%, and they are all below this value. We can say that these variables are significant in the prediction of AVG and have the most effect on the overall model.

We also wanted to look at how our model performed with respect to its residuals. We output a residual vs. fitted graph and were able to see that there was a good amount of variance in the fitted values compared to the reference zero line. In this model we had fairly homoskedastic data points, meaning constant variance of error term. This means we had the same variation of error throughout the model which is generally a good thing compared to heteroskedasticity. We can conclude that although we had some prediction error in our model, the error was constant throughout, so we are more accurate in those terms. We also output a histogram of the residuals which showed that the distribution of residuals was mainly concentrated in the center of the graph where zero residuals occur. This also indicated that our model was pretty accurate because most of the observations were at or around zero residuals.

Additionally, we decided to output the root mean squared error (RMSE) and goodness of fit (R^2) of the linear regression model to determine overall model performance. The RMSE for the train and test sets were 0.0244 and 0.0282, respectively. When calculating the RMSE we want the output to be as close to zero as possible. The RMSE for our model is very low and a low RMSE indicates a high accuracy. Additionally, our model is not overfit because the test RMSE is very close to the train RMSE. The R^2 of our model on the training and testing data was 0.499 and 0.486, respectively. When looking at R^2 values, we always want the highest value because a high R^2 means higher variation explained by the model and an overall better fit. In this case, both R^2 values are somewhat low as we want them to be as close to 1 as possible. We can say that the train and test data explained about 50% of the variation in the model.

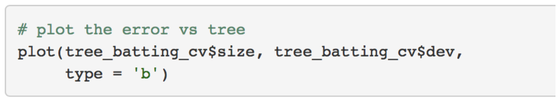
In conclusion, because we had very low RMSEs for both the train and test data and we had decent R^2 values we can say that our linear model was overall fairly accurate. Additionally, because of the heteroskedasticty of the model and the focus of distributions of residuals towards zero, we can conclude that error in our model may have been present, but it was consistent and not all over the place. When determining which variables are most important in predicting batting average, we look at the p-values associated with them. We can conclude that runs (R), doubles (X2B), runs batted in (RBI), walks (BB) and strikeouts (SO) are the most significant. A hitter should focus on these specific areas of the game if they are looking to increase their batting average. They should focus on increasing the amount of runs they score, increasing the amount of doubles they hit, increasing their RBIs, decreasing their walks, and decreasing their strikeouts. This is the optimal solution for hitters to make their batting average better.

Model 2: Decision Tree

For the second model, we chose to implement a decision tree to predict a hitter's batting average. The goal of the decision tree was to look at which variables would help a hitter increase their batting average the most. In the decision tree, the data is divided into splits, or true and false segments, and the chosen splits are ones which maximize separation of high and low batting averages. The root node that was output from the tree was Runs (R) which is our most important split. As we go down the tree, more splits are made until we reach the leaf nodes. At every leaf node, we take the batting averages from all our observations and take the average which gives us our prediction. So for example, if we added a new observation (test data) we could identify what region the hitter would fall into and make the prediction about what their batting average would be.

We wanted to figure out when to stop splitting the data because trees with many splits can overfit the data. To do this, we pruned the tree by growing a large tree, then penalized the size of the tree to create a sub-tree. The idea is that we pick the penalty that gives us the best cross-validation performance (lowest RMSE). In figure 2, we have plotted the tree size on the x-axis and the cross-validation (RMSE) on the y-axis. This plot gives us the best tree size at about 12-13 leaf nodes which gives us the lowest RMSE.

Figure 2: Tree size vs. RMSE



```
# plot the error vs tree
plot(tree_batting_cv$size, tree_batting_cv$dev,
     type = 'b')
```

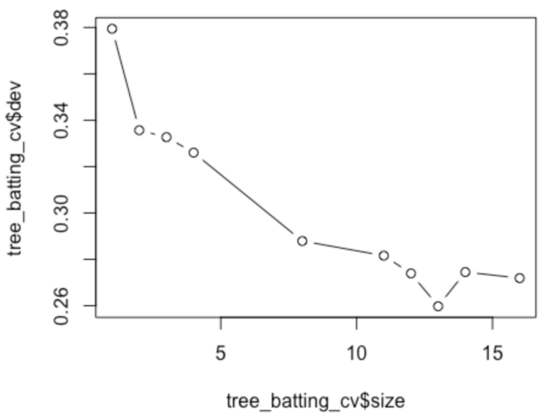
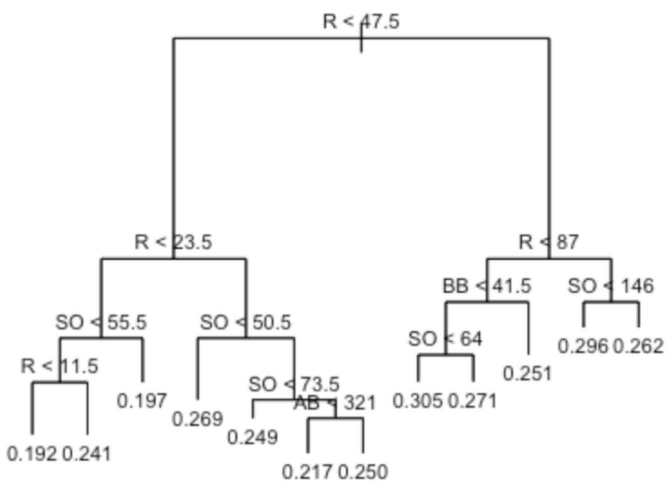


Figure 3: Pruned Tree



When looking at the pruned tree in figure 3, we can see that Runs is the most important split followed by strikeouts and walks. The leaf nodes give us the average batting averages for each split and, by following the tree path, a hitter can see what kind of statistics they should focus on to achieve their optimal batting average. Additionally, the RMSEs for the test and train sets were 0.022 and 0.031, respectively.

In conclusion, the 12 splits in the decision tree gave the lowest RMSE value to make our model as accurate as possible. Both the tree and pruned tree RMSEs were low and close to each other indicating an accurate model with minimal overfitting. Based off the pruned tree we can conclude that Runs is the most important split followed by strikeouts and walks. Going down the tree we can see that players with R > 47.5, R > 87, SO < 146 output the highest AVG of 0.296. Players with R < 47.5, R < 23.5, SO < 55.5, R < 11.5 output the lowest AVG of 0.192. Hitters should focus on these variables when looking to increase their batting average.

Model 3: Lasso Regression

For our 3rd model, we decided to implement lasso regression to determine the optimal value of lambda and to zero-out variables that the model finds to be insignificant. The variables G (games), AB (at-bats), R (runs), X2B (doubles), X3B (triples), HR (home runs), RBI (runs batted in), SB (stolen bases), BB (walks), SO (strikeouts) and HBP (hit by pitch) were used to predict a player's batting average (AVG). We set the alpha = 1 in order to create the lasso model. The graph in figure 4 shows the mean-squared error plotted on the y-axis, the log values of lambda plotted on the x-axis, and the numbers on the top of the graph represent the number of non-zero variables. The two vertical dashed-lines represent the lambda.min and the lambda.1se. The lambda values are the rate at which the coefficients are penalized. Lambda.min stores the lambda value that minimizes cross-validated error and lambda.1se stores the lambda value that minimizes cross-validated error plus one estimated error. At lambda.min, there are eight non-zero variables remaining and at lambda.1se, there are seven non-zero variables. We can also see that as lambda increases, the MSE does as well.

Figure 4: MSE vs Lambda

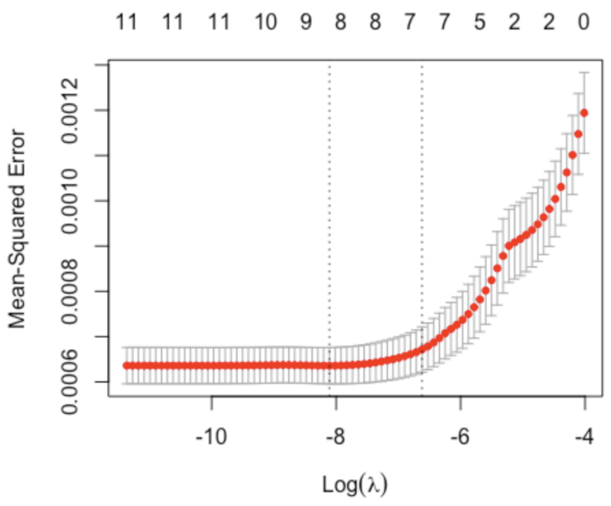
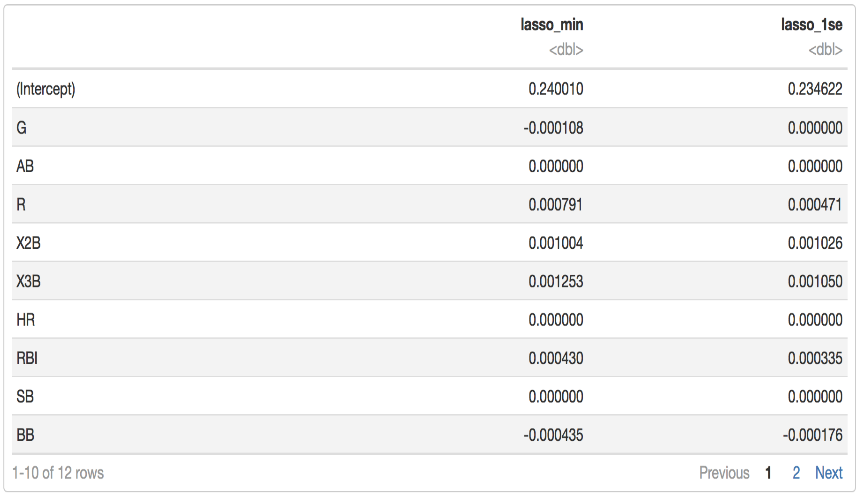


Figure 5: Lambda Values



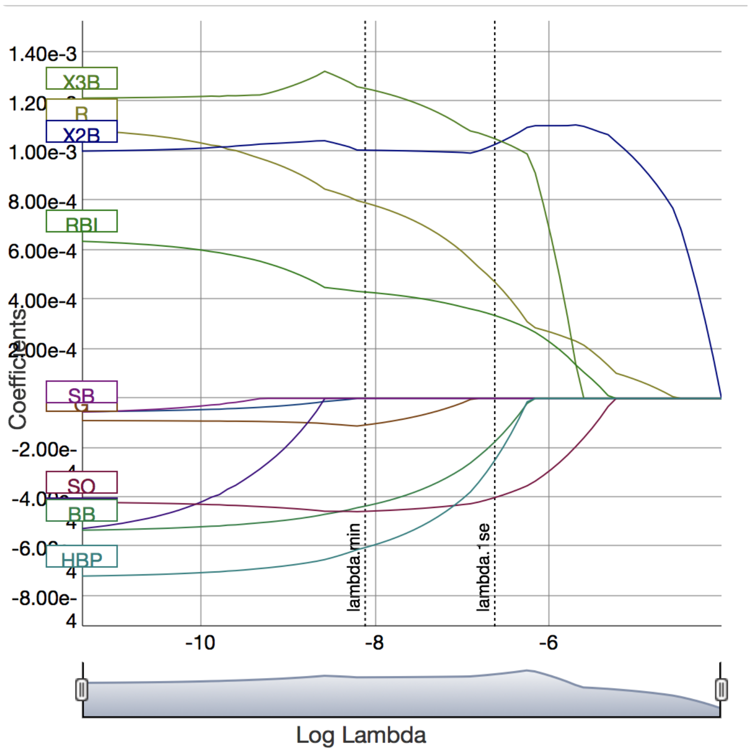
	lasso_min <dbl>	lasso_1se <dbl>
(Intercept)	0.240010	0.234622
G	-0.000108	0.000000
AB	0.000000	0.000000
R	0.000791	0.000471
X2B	0.001004	0.001026
X3B	0.001253	0.001050
HR	0.000000	0.000000
RBI	0.000430	0.000335
SB	0.000000	0.000000
BB	-0.000435	-0.000176

1-10 of 12 rows Previous 1 2 Next

After running the model, we printed the suggested lambda values in the graph in figure 5 to examine which coefficients have or have not been zeroed out. Variables are insignificant when they no longer have a value or their value has been converted into a period. When using lambda.min, the three variables AB, HR, and SB, were zeroed out and had no significance. For lambda.1se, AB, HR , SB, and G were zeroed out. Just as we saw in the plot, one more variable was zeroed out in lambda.1se. We were surprised to see the model zero-out HR (home-runs), as you would think the amount of home-runs a player would have would significantly impact their batting average. It is much easier to understand why the variables AB, SB, and G. As we mentioned earlier in the project, a great universal batting average percentage is around .300. In other words, out of ten at-bats, a great batter would get a hit three times. The odds of hitting a home-run are much smaller than that so maybe that is why the lambda.min decided to zero out that variable.

Figure 6 visualizes the coefficient shrinkage path of what values of lambda the coefficients zero out or make insignificant. At the optimal values of lambda (lambda.min and lambda.1se), the cross-validated error is minimized and the MSE error is reduced. At lambda.min, the variables AB, HR, and SB were shrunk. At lambda.1se, G is zeroed out. Between the optimal values of lambda, the variables R, H, X2B, X3B, RBI, BB, SO, and HBP were still significant predictors of a player's batting average.

Figure 6: Coefficient Shrinkage Path



In conclusion, in order to find insignificant variables and select the important ones, the Lasso model zeroed-out a range of variables. It eliminated games, at bats, homeruns, and stolen bases. This means we can now interpret important variables for variables selection. The variables that the model selects as significant include runs, doubles, triples, runs batted in, walks, strikeouts, and hit by pitch. A hitter can now realize that hitting home runs and getting more stolen bases doesn't necessarily impact their batting average.

Project Conclusions

Out of the three models our group created, we decided to implement the decision tree model as it would provide the best use case for MLB hitters. It would make the most sense logically for hitters as we can share this model with them and they can simply examine the tree and look at which variables would improve their batting average.

When compared to the other two models, the decision tree model output the lowest RMSE value indicating the most accurate model. The decision tree model is also the easiest to interpret out of the other two models as MLB hitters can easily identify what areas would improve their batting average, whereas in the linear regression model it might be more difficult for an MLB hitter to understand. With our results, we could present them to the owners and managers of MLB teams to give them a competitive advantage over their opponents. This would help them achieve their ultimate goal of achieving a World Series championship.