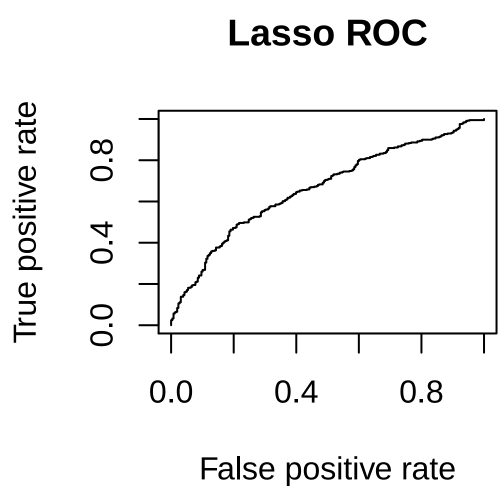
Project Part Two: UFC Fight Predictor

Cary Dean Turner and Tony Kim

Part One: Prediction on the Test Set

Classification

In predicting the fight outcome on our test data set, our results were fairly consistent with our estimates from cross-validation. Our cross-validated estimate of AUC was 0.706, while on the test data our chosen model achieved an AUC of 0.6631—slightly under our estimate. As can be seen in the table below, they also had somewhat similar values for Accuracy, Precision, and False Discovery Rate. The most notable differences in performance are in Sensitivity, where the model performed better on the training data, and specificity, where the model actually performed notably better on the test data. The Type 1 error rate was also lower on our test data, which is somewhat surprising.



Lasso Confusion Matrix

```
##
## lasso.lr.pred.test   0   1
##                  0 147 127
##                  1 111 242
```

Lasso Prediction Metrics on Training and Test Data

```
##                  Metric     Train      Test
## 1              Accuracy 0.6590909 0.6204147
## 2              0-1 Loss 0.3409091 0.3795853
## 3           Sensitivity 0.7588358 0.6558266
## 4           Specificity 0.5239437 0.5697674
## 5             Precision 0.6835206 0.6855524
## 6     Type I Error Rate 0.4760563 0.4302326
## 7    Type II Error Rate 0.2411642 0.3441734
## 8  False Discovery Rate 0.3164794 0.3144476
```

```
## 9                CV AUC 0.7068651        NA
## 10             Test AUC        NA 0.6639146
```

Regression

Tony got dis

Part Two: Inference

Tony got dis shit too

Coefficients on Training Data

```
##                 Estimate   Std. Error    z value     Pr(>|z|)
## (Intercept)  0.055107859 0.0477957404   1.152987 2.489158e-01
## R_odds      -0.002661767 0.0001974061 -13.483712 1.950413e-41
## age_adv     -0.039914177 0.0089609559  -4.454232 8.419401e-06
```

After fitting our chosen model and performing inference on the training data, we then re-fit the same model (logistic regression using R_odds and age_adv) onto the held out test data. The results were noticeably different. As noted above, when the model was fit on the training data, we saw that both covariates (R_odds and age_adv) were statistically significant at the 0.001% level, but when the model is fit on the test data we see that, although R_odds remains statistically significant at the 0.001% level, age_adv no longer appears to be significant, with a p-value of 0.345. This could be due to the fact that when looking at the results from the training data, our results are subject to post-selection inference. This is because when we are performing inference, we are using the model which was chosen by lasso on the training set, so our results are biased in favor of those covariates that the lasso selected. However, it's also worth noting that, because our training data set is so much larger than the test data set, it's likely a better representation of the true underlying population, the inference results on our test set, although unbiased, may have higher variance.

Coefficients on Test Data

```
##                 Estimate   Std. Error    z value     Pr(>|z|)
## (Intercept)  0.123304265 0.0949777820  1.2982433 1.942038e-01
## R_odds      -0.002313613 0.0003935162 -5.8793339 4.119207e-09
## age_adv     -0.016913755 0.0178959208 -0.9451179 3.445987e-01
```

Confidence Intervals

In performing the bootstrap to obtain confidence intervals, our results were almost identical, which can be seen in the tables below. We chose to use the normal interval in this case because all of our coefficient estimates has distributions that were very close to normal. We then performed the bootstrap to compute intervals on the test data and similarly got confidence intervals that were shockingly close to the ones computed by glm(). One additional thing to note is that our confidence intervals computed on the test data were almost exactly twice as large as the confidence intervals computed on the training data, which is consistent with the fact that the training data set is four times as large as the test data, and the fact that standard errors are proportional to 1/sqrt(n).

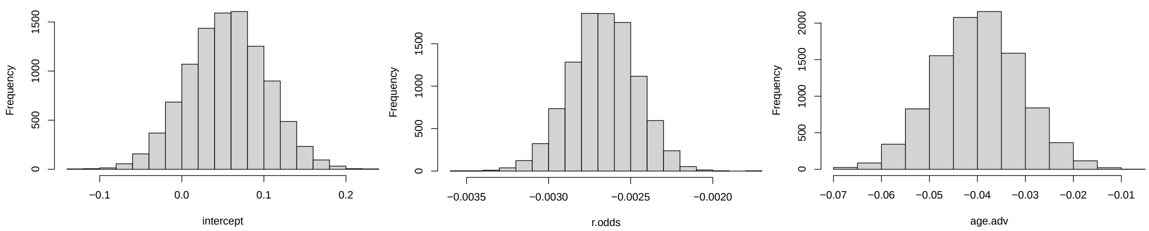
Normal Confidence Intervals from glm()

```
##                       Low         High         Size
## (Intercept) -0.038571793  0.148787510 0.1873593024
## R_odds      -0.003048682 -0.002274851 0.0007738318
## age_adv     -0.057477650 -0.022350703 0.0351269472
```

Normal Confidence Intervals via the Bootstrap

```
##                     Low         High         Size
## (Intercept) -0.038910221  0.149125938 0.1880361593
## R_odds      -0.003054808 -0.002268725 0.0007860821
## age_adv     -0.057381074 -0.022447279 0.0349337948
```

Bootstrapped Distribution of Coefficients from Training Data



In an effort to be more conservative in our p-value estimates, we performed both the Benjamini-Hochberg and Bonerroni processes on the p-values produced by glm(). Neither of the two processes increased the p-values of our significant coefficients enough that they became insignificant. Both R_odds and age_adv remain statistically significant at the 0.01% level.

P-Values from Training Data

```
##            (Intercept)       R_odds      age_adv
## GLM Output   0.2489158 1.950413e-41 8.419401e-06
## BH           0.2489158 5.851240e-41 1.262910e-05
## Bonferroni   0.7467474 5.851240e-41 2.525820e-05
```

P-Values from Test Data

```
##            (Intercept)       R_odds   age_adv
## GLM Output   0.1942038 4.119207e-09 0.3445987
## BH           0.2913056 1.235762e-08 0.3445987
## Bonferroni   0.5826113 1.235762e-08 1.0000000
```

We separately fit a model on the training data using *all* the non-transformed covariates (i.e., not just the ones that lasso selected). When doing this we found that both R_odds and age_adv remained significant at the same level, but we also saw that several other variables (TD_landed_adv, reach_adv, and win_streak_dif) also came up as statistically significant at the 5% level. This tells me that perhaps these values are likely to be non-zero (statistically significant), but not practically significant (i.e., they didn't reduce prediction error significantly, so the lasso zeroed them out). However, after running BH and Bonferroni on these p-values, the only ones that remained significant are R_odds, age_adv, and TD_landed_adv, which is very close to the subset of variables selected by lasso (R_odds and age_adv).

P-Values on Coefficients in Model Using All Covariates

```
##                  R_odds      age_adv  reach_adv TD_landed_adv win_streak_dif
## GLM Output 1.585986e-31 0.0001800311 0.02562526  0.0002758013     0.02287974
## BH         6.026746e-30 0.0034205911 0.19475195  0.0034934837     0.19475195
## Bonferroni 6.026746e-30 0.0068411823 0.97375977  0.0104804511     0.86942995
```

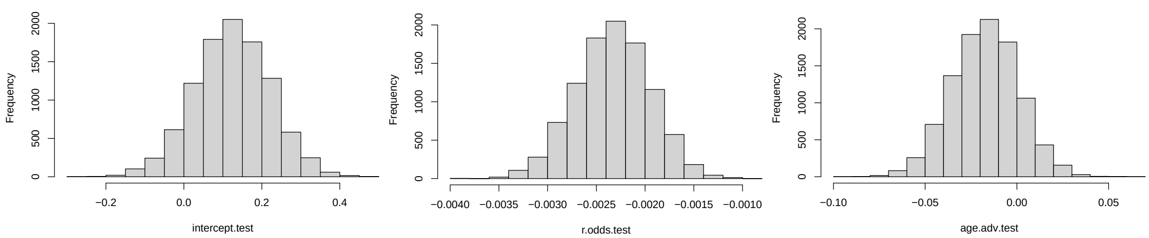
Part Three: Discussion

The main real-life application of our models would certainly be in the context of gambling. Bettors are always looking for new and improved ways to beat the odds, and this could certainly be useful for that. For example, if the Vegas gambling odds favor a particular fighter but our model strongly suggests a different outcome, this could produce a lucrative opportunity for bettors to get one over on Vegas. The prediction of fighter odds could similarly be used to discover when Vegas odds are over or under what they should be, also resulting in potentially lucrative opportunities for bettors.

One other potential application, however, could be for fighters, managers, and promoters. For example, fighters and managers could use these models to pick which fights they want to take. Nobody wants to take a match that they have a projected 90% chance of losing. Similarly, promoters generally like to pit fighters against other fighters of equal skill, as it results in a more interesting and entertaining match for the fans. These models could be used to find potential fights with close to 50/50 odds which might be better fights and hence earn more money via ticket sales and pay-per-views.
Although our models were trained on several thousand observations, it would almost certainly be a good idea to update them regularly for two reasons: The first is that our predictive models, although respectable, still leave a lot of room for error—more future data will likely help remove some of that error. The second reason is that the sport of MMA is still relatively young and constantly evolving, meaning the attributes that favor a fighter today may be completely different than the ones that favor a fighter in five or ten years.

Appendix / Extra Shit

Bootstrapped Distributions of Coefficients on Test Data



glm() Confidence Intervals on Test Data

```
##                     Low         High        Size
## (Intercept) -0.062852188  0.309460718 0.372312905
## R_odds      -0.003084905 -0.001542321 0.001542583
## age_adv     -0.051989760  0.018162250 0.070152009
```

Bootstrapped Confidence Intervals on Test Data

```
##                     Low         High        Size
## (Intercept) -0.065712999  0.312321529 0.378034527
## R_odds      -0.003064909 -0.001562318 0.001502591
## age_adv     -0.052481418  0.018653908 0.071135327
```