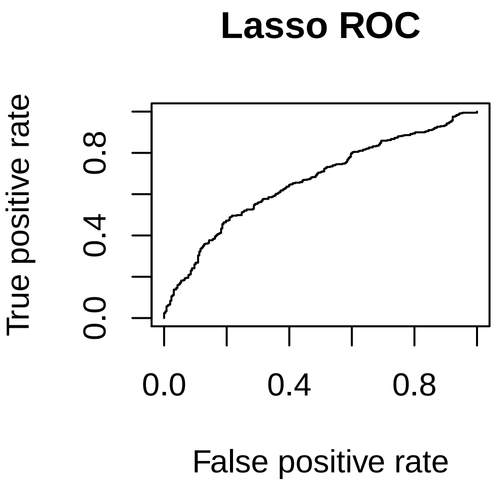
Project Part Two: UFC Fight Predictor

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 0.663—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

In predicting the red fighter's odds on our test data set, the results are again fairly consistent with our estimates from cross validation. The best model from cross validation was a lasso model that used transforms and interaction terms, and its cross validation error was 226.448. Our estimate for the test error from Part 1 was 229, as a conservative estimate that the test error would be slightly higher than the cross-validation error. The test error actually turned out to be lower than the cross-validation error, with its value as 225.247. This is very reasonable because our analysis in Part 1 also mentioned that because our number of cross-validation folds k=10 is a relatively small number, the cross validation could certainly overestimate the test error; the use of less data in each fold may have introduced more bias. All in all, the estimate was very close to the real test error.

As seen in the Appendix, we have also attached plots of residuals against fitted values and residuals against covariates such as height and weight advantage. All 3 plots show a relative lack of correlation between residuals and fitted values / covariates, indicating that the linear model is a good estimate of the underlying population model.

Part Two: Inference

For our analysis, we choose the logistic regression model that predicts whether the Red fighter wins. We perform our analysis by fitting a generic logistic regression on the reduced set of covariates given by our lasso model in Part 1, which in our case is two variables: (1) R_odds [the odds of the Red fighter] and (2) age_adv [the age difference between the two fighters]. As we can observe in the table below, both coefficients are statistically significant, with each of their p-values being less than 1e-05. This means that given that the null hypotheses that each coefficient is 0 are true, the probability of observing coefficients as extreme as each of our fitted model's coefficients is less than 1e-05 (hence very unlikely). Note that this does not equate to practical significance, since the relationship between these 2 covariates and the outcome can be weak. We cautiously believe these results, since it would make sense that pre-match odds and age difference in fighters have nonzero effects on the match's outcome. At the same time, we are aware of possible biases such as post-selection inference that downgrade the quality of these results, as we will detail below.

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 on the training set, we are using the model which was chosen by lasso on that same training set, so our results are biased and overly optimistic 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 distribution interval in this case because all of our coefficient estimates have distributions that were very close to normal. We then performed the bootstrap to compute intervals on the test data (see Appendix) 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).

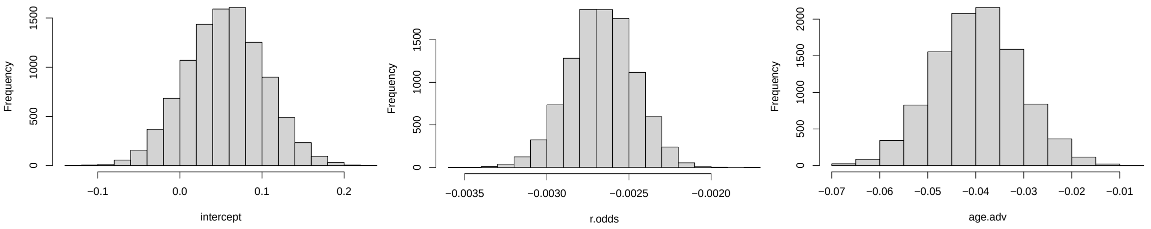
Normal Confidence Intervals from glm()

```
##                      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
```

We also investigated fixing the issue of post-selection inference mentioned above. We do this by splitting our entire data into two equal parts (a 50-50 ratio) and running lasso to select covariates on the first split. Then, after fixing the model to our selected covariates, we fit a generic logistic regression on the unseen second split of data to determine significance. In this case, the lasso model selects the R_odds coefficient only, and when performing inference on the unseen data, the R_odds coefficient is identified as statistically significant, with its p-value at the level of 1e-36. Thus, we confirm through this method that the R_odds coefficient is statistically significant even in a more relatively unbiased setting that mitigates the issue of post-selection inference. Additionally, this means that coefficients such as age_adv that showed statistical significance in earlier settings may have been prone to biases from post-selection inference; however, because the training data (80% of dataset) is a larger dataset than the 50% split that we use currently, it is possible that the training set represented true relationships more accurately between the variables in the population.

Coefficients and Confidence Intervals when Correcting Post-Selection Inference

```
##                 Estimate   Std. Error     z value     Pr(>|z|)
## (Intercept) -0.010218302 0.0579576840  -0.1763063 8.600533e-01
## R_odds      -0.002971806 0.0002362682 -12.5781056 2.786318e-36
```

```
##                      Low        High         Size
## (Intercept) -0.123815363  0.10337876 0.2271941213
## R_odds      -0.003434892 -0.00250872 0.0009261712
```

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.

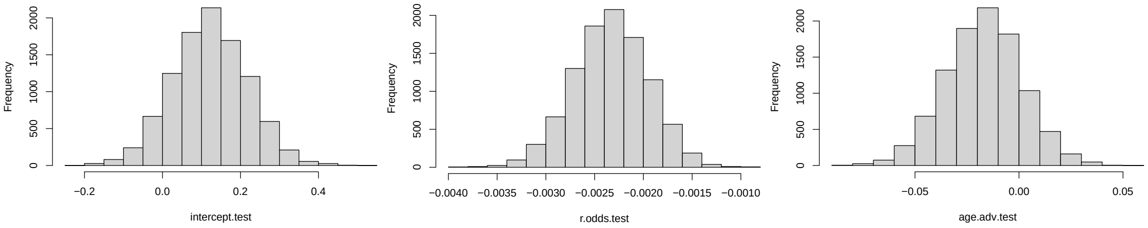
Anyone who uses our models should be aware of our method of transforming the data where the statistics of the two fighters in each match were collapsed into single statistics, each capturing the differences between the fighters. While these single combined statistics did a good job of explaining which player had the advantage, the degree to which these differences can explain the outcome must be taken with caution. For example, the same 4-year difference can have a different scale of effects depending on the fighters' ages; the difference between a 23-year-old and a 27-year-old is likely to mean less than the difference between a 31-year-old and a 35-year-old, as athletes start a sharper performance decline in their early thirties. With regards to overfitting, our best models do well in mitigating the problem. When our best models for both the classification and regression problems were chosen as the lasso model, both cases were able to eliminate unnecessary coefficients, and the test set prediction results had similar magnitudes of accuracy and error to those of cross validation. On the flipside, if a user wishes to utilize our other (non-lasso) models, they could potentially experience problems of overfitting, as these sub-optimal models seem to generalize less well in terms of having higher cross-validation errors in relation to training errors. Furthermore, it would be helpful to a user of this model to understand the effect of post-selection inference on our models; through our analyses, we determine that R_odds is the sole covariate that can be consistently statistically significant after applying corrections to post-selection inference. Thus, we are confident that R_odds has the most explaining power out of all potential covariates in whatever model the user chooses to run. Lastly, the user does not need to worry about the problem of multiple hypothesis testing, as we are only dealing with 1 or 2 hypothesis tests at the same time.

There is not much we would change about the data collection process; the dataset already kept track of a whopping 127 columns, so all the covariates we would have liked to have were present. The only real drawback with the data was the amount of missing values we were forced to delete, but otherwise it would be difficult to get a more detailed and robust dataset than what we already have.

In terms of attacking the same dataset again, we could choose to improve upon our methods of transforms. As mentioned before, our combining of the two fighters' statistics into differences between them may not provide the entire story. One idea to try would be to add statistics of the Red fighter in addition to the differences between the two fighters. This could potentially help in capturing the differing scale of effects in age in the examples of 23- vs 27-year-olds and 31- vs 35-year-olds mentioned earlier. Furthermore, while we tried applying cubic transforms and two-way interaction terms, we could explore further possibilities such as using logarithmic transforms that could help explain non-linear patterns.

Appendix

Bootstrapped Distributions of Coefficients on Test Data



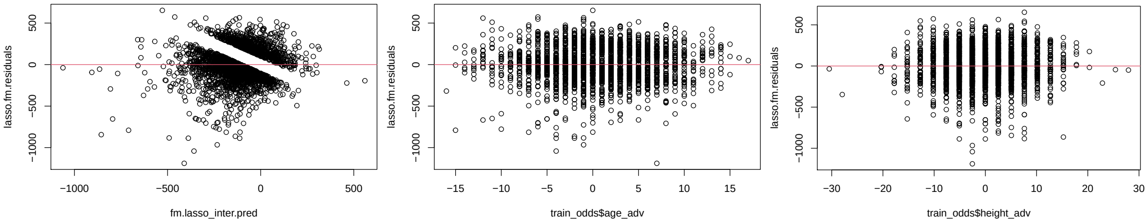
glm() Confidence Intervals on Test Data

```
##                     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) -0.064652687  0.311261217 0.37591390
## R_odds      -0.003061743 -0.001565483 0.00149626
## age_adv     -0.052880967  0.019053457 0.07193442
```

Residual Plots for Regression


lasso.fm.residuals
fm.lasso_inter.pred
lasso.fm.residuals
train_odds$age_adv
lasso.fm.residuals
train_odds$height_adv