Applying Machine Learning Algorithms to Predict UFC Fight Outcomes

McKinley McQuaide
mcquaide@stanford.edu

CS229: Machine Learning
Autumn 2019

Motivation

Sports betting is a $155 billion industry. Fighting ranks among the top in the industry, and the Ultimate Fighting Championship (UFC) is currently taking steps to push it even further. My goal was to explore our ability to predict the outcome of UFC fights based on the each match's pre-fight statistics using Machine Learning. An accurate prediction model could both inform the best placed bets (and potential risk associated) for each fight, but also could provide insight to coaches when accepting fights to begin with.

Data and Data Processing

There are over one hundred different fighter statistics on UFCStats for each of the 5,144 fights in UFC record history extending from 2019 back to 1993, which include information such as fighters' height, weight, reach, and stance, as well as statistics such as win streaks, strike percentage, guard passes, and strikes landed by location. The dataset used in this study was scraped from UFCStats and statistics pertaining to the circumstances of the fight (i.e. location, number of allocated rounds, etc.) were removed to as they were deemed irrelevant and did not align with the fighter-centric intention of the study. The set was then cleared of samples with incomplete feature sets leaving **3,355 complete samples** spanning from 1997 to 2019 with **134 features**.

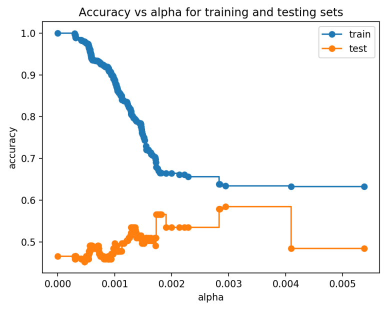
The testing and training sets were split using **k-fold validation** for **time-series** model with k=20 folds across the 21 years to determine a more representative accuracy of future predictive ability. Within each fold, the training data inputs were centered and scaled, and the scale was then applied to the test set inputs before fitting.

Classification Models

Four classification models[1] were used in this study:

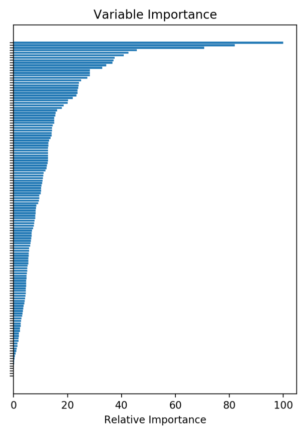
1. **GLM: Stochastic Gradient Descent** was applied with a perceptron loss function
2. **NN: Multilayer Perceptron**[2] was used with L-BFGS to optimize the log-loss function. Here a grid search was preformed to find the optimal layer size.
3. A **Decision Tree** was applied with minimum cost-complexity pruning with measured effective node alpha as follows[3]:

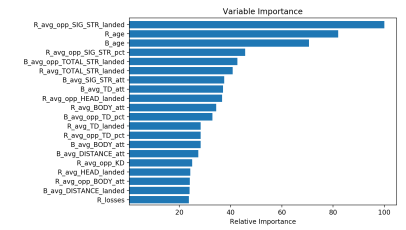
$$R_\alpha(T) = R(T) + \alpha|T| \qquad \alpha_{eff}(t) = \frac{R(t)-R(T_t)}{|T|-1}$$

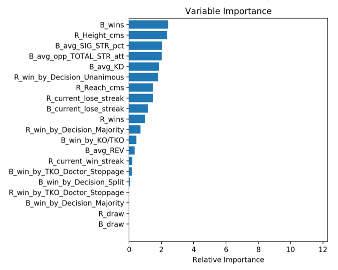


complexity parameter identified at ccp α = .003

4. **Gradient Boosting** was applied with a multinomial deviance loss function and a learning rate of 0.01, producing the following relative variable importances:







Prediction and Results

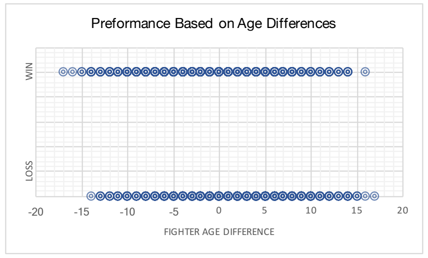
After accounting for overfitting and creating the models, the accuracy was computed averaging over the k-folds using **Mean Squared Error:**

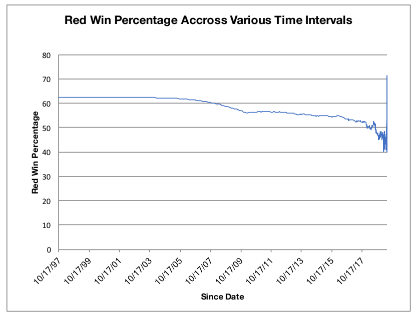
$$MSE = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2$$

	Average Training Accuracy (final fold: 3187)	Average Testing Accuracy (k * 168)
Stochastic Gradient Descent	72.876%	58.018%
NN: Multilayer Perceptron (L-BFGS)	89.566%	57.736%
Decision Tree (ccp α = .003)	80.222%	60.252%
Gradient Boosting	88.583%	61.226%

Discussion

Overall, Gradient Boosting seemed to preform the best, however the average test accuracies all land around 60%. Observing the variable importances, correlations were explored between simply being the red fighter and winning. While the entire dataset, the red fighter wins 62.6% of the time, looking at the red fighter win percentage over time highlights the change in trends and supports using incremental training sets via time series k-fold. Most interesting is that consistently, fighter age was one of the highest weighted features.





Future

Moving forward, the parameters should be further adjusted to improve the accuracy of predictions, continuing off of the initial grid search, perhaps replacing it with a randomized search, and applying similar functions in other areas of the study. Additionally, if a pass is taken removing irrelevant parameters, then previously removed samples may be added back in as they may become complete. Repeating this would enable more of the data to be used.

References

[1] *Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.*
[2] *Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).*
[3] *L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth, Belmont, CA, 1984.*