MMA Bout Outcome Prediction

Vaughn Franz
Northeastern University
vaughnfranz@gmail.com

1 Introduction and Data Description

The sport of MMA (Mixed Martial Arts) is relatively under-studied from an analytic perspective. Compared with other major sporting organizations there is a lack of advanced statistics and what statistics are available are often regarded as unreliable. In part, this is likely due to its relative youth, its relative obscurity and the fair degree of stigma which still surrounds it. Although, in part it it likely due to the nature of the competition as well. The outcome of matches, when they do not end in a KO (knockout), TKO (technical knockout), or submission, are determined by a panel of three judges. What constitutes a successful or effective application of technique in striking or grappling has some degree of definitional ambiguity.

All of the factors which make the acquisition of accurate and meaningful statistics difficult also make the prediction of results difficult. To make matters worse, the bouts have a variable length. They can be contested over either three or five 5-minute rounds. Fans and analysts alike widely consider the sport to be unpredictable. The possibility of a knockout or submission scored in the waning seconds of a match in which one fighter has been dominating have been analogized to a hypothetical 100-point shot in basketball which could instantaneously erase a large deficit.

The task of predicting bout outcomes in MMA is one which is of interest to fans, athletes, sports organizations and bettors alike. In this project, I attempt to use some of the best publicly available statistics to build machine learning models capable of predicting the outcomes of bouts.

1.1 Data Source

The data was obtained from kaggle [1]. The data was originally scraped from ufcstats.com, which is a site unaffiliated with the UFC housing unofficial statistics.

1.2 Data Content

The data has records of 6,012 bouts having occurred in the UFC between 1993 and 2021. Each record in the table has 144 features. Each record has a information relating to a "Red" fighter or a "Blue" fighter (the red and blue labeling of the corners is a convention followed in many combat sports) along with a column designating the winner of the bout.

It is important to note that the red and blue distinction is not arbitrary. The fighter in the red corner is typically the "favorite," the champion or otherwise well known. This information has some bearing on the problem as is discussed later in the paper.

An exhaustive description of each column is not provided here but a brief description follows.

Physical attributes of each fighter are provided (height, reach, weight etc.). Statistics about each fighter's previous wins and losses (number and outcome) are provided.

A number of descriptions of the bout characteristics are provided (location, date, title or non-title, referee name, fighter names).

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

The statistics relate to each fighter's striking and grappling output and defense. For striking the statistics are broken down into attempted and landed. They are provided for various regimes of the fight (e.g. at distance, in the clinch, on the ground) as well as for various targets (e.g. head, body, leg). For grappling the statistics focus on takedowns (attempted and landed), reversals (reversing position from bottom to top or non-dominant to dominant), and submission attempts. The defensive complement of these statistics is also provided for each fighter. To take one example, the average number of submission attempts made by a fighter and against said fighter per minute are recorded.

1.3 Data Processing and Problem Formation

A number of steps were taken to clean and prepare the data.

1. the Winner of the bout is recorded as either "Red," "Blue," or "Draw." The draws were rare, and given the size and nature of the dataset I decided to remove all records of fights which ended in a draw. This made the problem into a binary classification problem where the label ("Red" or "Blue") is to be predicted for each row in the data.

2. The following columns were removed from the dataset before training: red fighter name, blue fighter name, referee name, date, location, title bout (boolean), weight class (a string), number of draws on blue's record, and the number of draws on red's record. These were removed for presumptive lack of predictive power.

3. Some missing values were present for the reach of a fighter (distance from fingertip to fingertip with arms outstretched to the side). Reach is correlated with height in humans at an average ratio of 1:1. So, where missing the reach was replaced with the height.

4. Some values were missing for the stance of a fighter (orientation while striking based on handedness). These were replaced with the modal value indicating a right-handed stance ("Orthodox").

5. The columns representing the stance were the only categorical columns remaining in the dataset. These were one-hot encoded to make them numerical. There were a total of 4 categories resulting in 8 total columns.

6. The data was then split into testing and training datasets. Twenty percent of the data was set aside for testing. The testing and training data were created to have roughly equal proportions of "Red" and "Blue" labels.

After splitting the data the following steps were completed.

1. Missing values for numerical columns were imputed using the median of the column. This was done using the `SimpleImputer` from `sklearn`. The amount of missing values was significant, over 1000 rows missing for certain columns. The likely cause of this is the fact that fighters making their debut do not have a previous record of fights from which to draw statistics. For this reason also, the missing values are much more prevalent for the "Blue" fighter.

2. The numerical columns were then scaled using `RobustScaler` from `sklearn`.

3. The same `SimpleImputer` and `RobustScaler` which were fit to the training data were then applied to transform the testing data.

The results of this process was a testing set with 4721 rows and 143 features. The testing set had 1181 rows and again 143 features.

2 Models and Training Procedure

2.1 Models

Four models were trained on the data, all coming from `sklearn` modules.

1. `LogisticRegression` - a basic logistic regression
2. `LogisticRegression` - a basic logistic regression with an L2 penalty term
3. `RandomForestClassifier` - an ensemble method which fits a number of decision trees to the data and aggregates the prediction over them
4. `MLPClassifier` - a fully connected neural net with one hidden layer

2.2 Training and Hyperparameters

For the logistic regression models the only hyperparameter used is the number of iterations. In each case the regressors were trained for a maximum of 3000 iterations. The classifiers were simply trained on the training data set.

For the `RandomForestClassifier` and the `MLPClassifier`, some hyperparameters were chose using `GridSearchCV` from `sklearn`. This class allows for iterative testing of all combinations of user provided values for optimal cross validation accuracy.

The `RandomForestClassifier` was tuned across 10, 100 and 1000 possible estimators (number of trees in the forest) and a maximum depth of 10, 100 or None (This is the maximum depth of a tree in the forest; None corresponds to no limit). The grid search was done on 5 cross validation folds. The final parameters for the tuned model were 1000 estimators and a max depth of 100. The cross validation was done on the training data set.

The `MLPClassifier` was tuned across various sizes of a single hidden layer (1, 2, 5, 10, 50, 100), and two choices of activation function (`relu` and `tanh`). The grid search was again done on 5 cross validation folds. The final parameters for the tuned model were a hidden layer size of 2 neurons and a `tanh` activation function. The cross validation was done on the training data set.

3 Results

For each model two ROC curves and two precision-recall curves are plotted using the data set aside for testing. The reason for 2 curves each is because I wanted to look at the performance considering either the "Red" or the "Blue" to be the "positive" class when considering the performance.

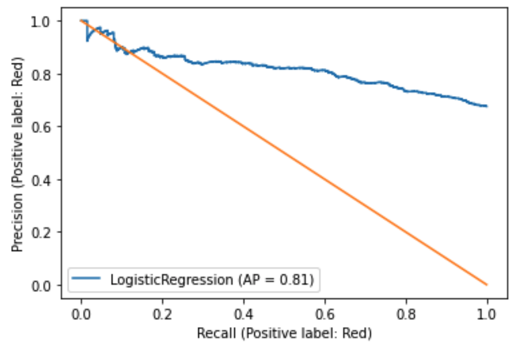
The performance of the two logistic regression classifiers was nearly identical. I believe that this is because a large number of the features had some positive performance on the predictive power of the model. This was further evidenced by a calculation of ANOVA scores for all features which showed fairly high f-scores (>10) for a big majority of the features.

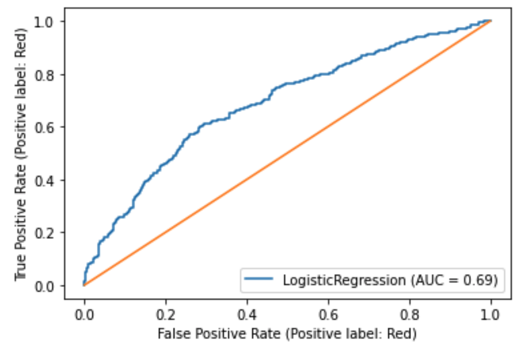
All things considered the logistic regressors performed the best out of all of the models which were trained. They had the highest area under the curve on the ROC curves, as well as higher average precision on the precision-recall curves.

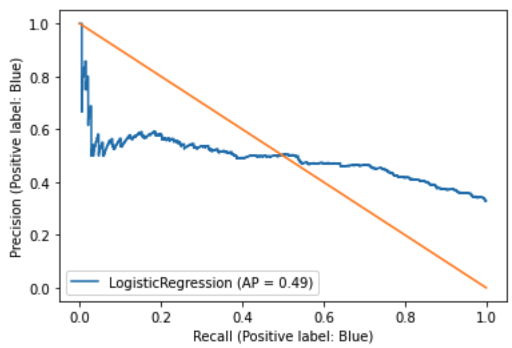
The random forest classifier performed the worst. It had the lowest area under the curve and the lowest average precision. Considering the very small size of the hidden layer selected via cross validation, this could indicate that a neural net may not be the best choice for solving this problem as it stands.

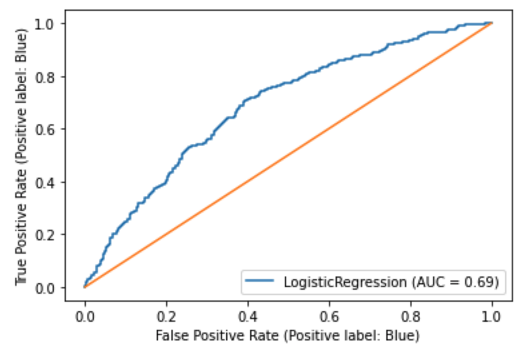
All classifiers performed poorly in correctly categorizing "Blue" victories. The issue of missing data considered earlier could be a contributing factor.

3.1 Logistic Regression No Penalty

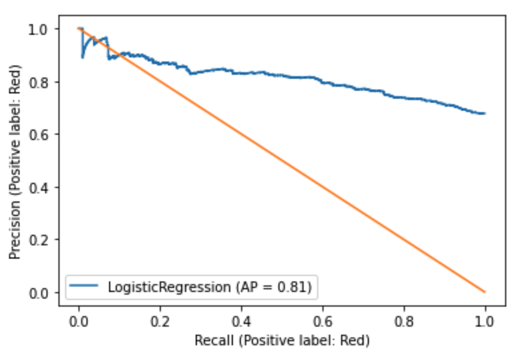


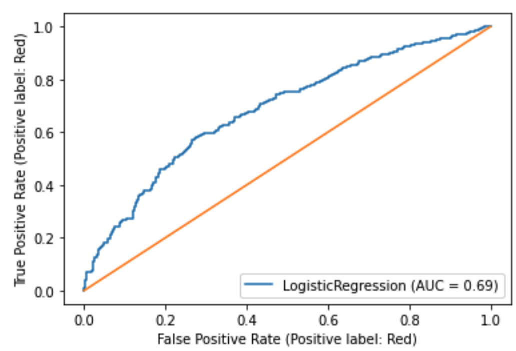


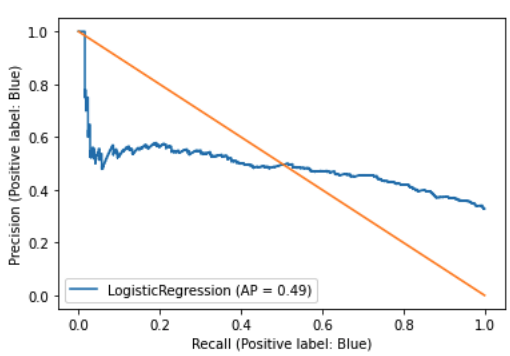


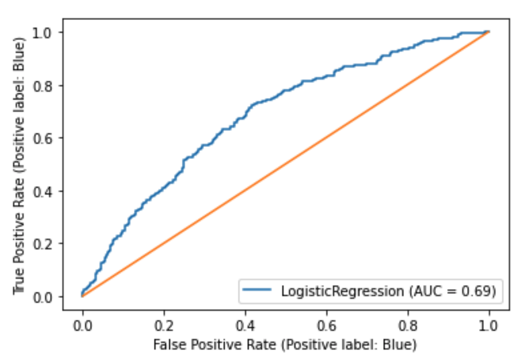


3.2 Logistic Regression L2 Penalty

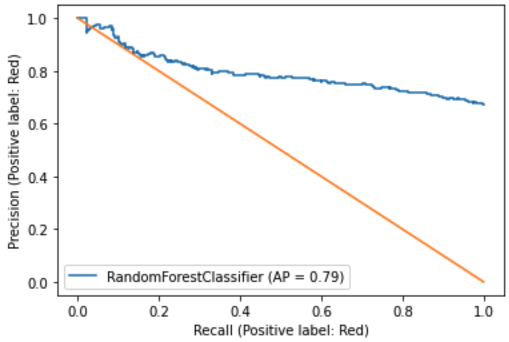


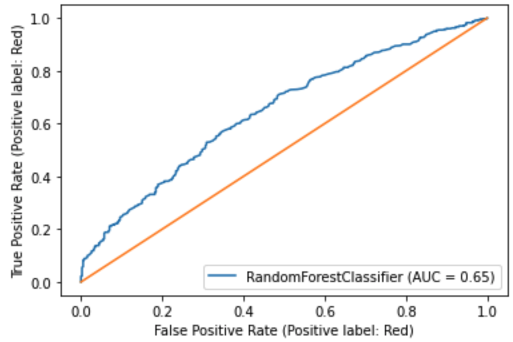


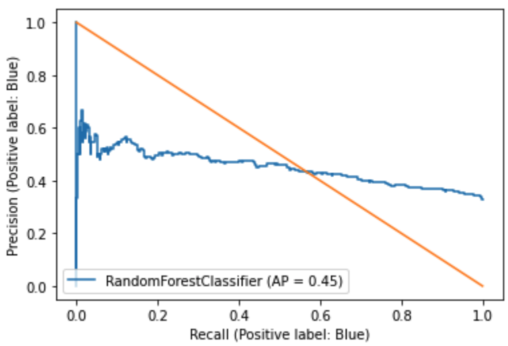


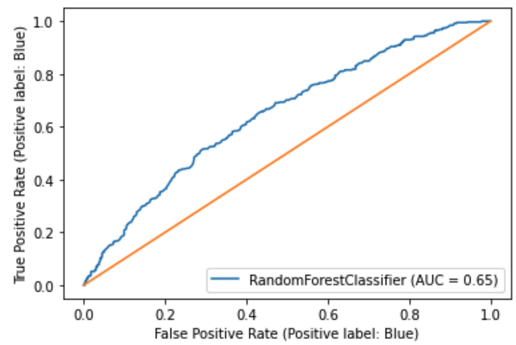


3.3 Random Forest Classifier

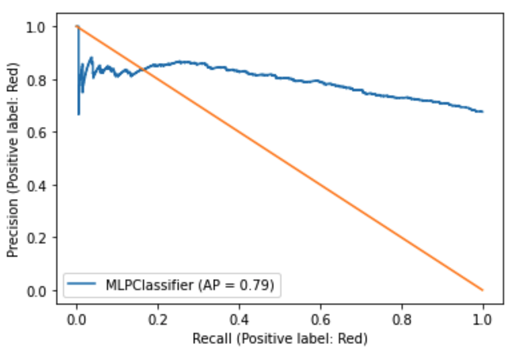


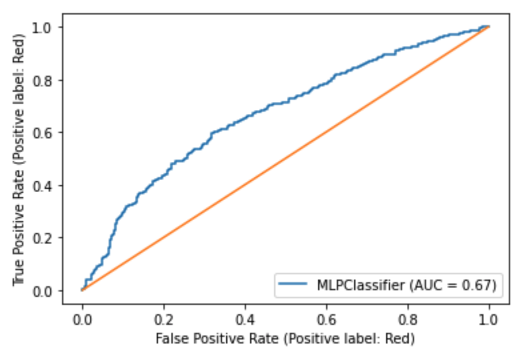


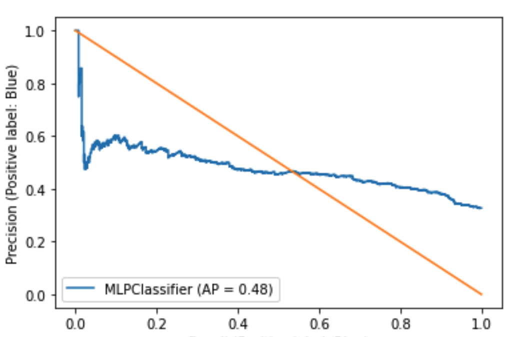


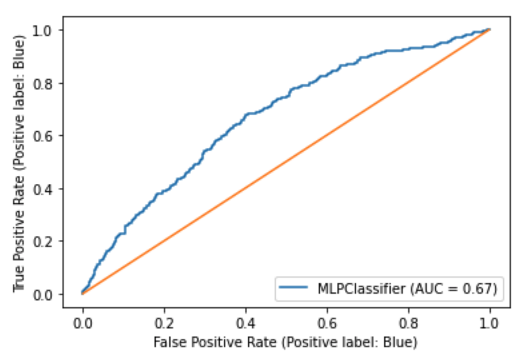


3.4 MLP Classifier









References

[1] https://www.kaggle.com/rajeevw/ufcdata

[2] https://scikit-learn.org/stable/modules/generated/sklearn.linearmodel.LogisticRegression.html

[3] https://scikit-learn.org/stable/modules/generated/sklearn.neuralnetwork.MLPClassifier.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[5]https://scikit-learn.org/stable/modules/generated/sklearn.modelselection.GridSearchCV.html