Machine Learning in UFC Prediction

Ilan Barr

Washington University in St. Louis

Advisor: Dr. Cynthia Ma

April 24, 2023

1 Introduction

Perhaps the aspect of being a sports fan that provides the biggest appeal is the fact that the matches are unpredictable. While one can make predictions on the outcome of an event, no one truly knows who the victor will be until it is over. This unpredictability, however, doesn't deter people from believing they are savvy enough to make accurate predictions on a variety of sports. Through the avenue of sports betting, these individuals have an opportunity to test their knowledge, potentially profiting from their expertise. The harsh reality, however, is that the sportsbooks setting the odds generally have the advantage because of their extensive resources, algorithms, and ability to adjust the odds due to new information and betting patterns. Due to the bookmaker's margin, or the vig, if a sportsbook accurately predicts, or even comes close, the probability of an event occurring, such as who will win a football match, it makes beating them impossible.

Sportsbooks aim to set accurate odds to ensure profitability, but inefficiencies in the market can sometimes be exploited by savvy bettors who can identify when the odds do not reflect the true probabilities of the outcomes. Some sports betting markets are notoriously more efficient than others, for example NFL spreads at the start of the game are virtually unbeatable[1] , where in the long run, betting on either team will be a losing strategy.

The goal behind this project was to research, design, and implement a machine learning model that would be able to outperform sportsbooks in predicting sporting events. After doing background research into a few potential sports, I landed on the UFC as my target league. There has been little quantitative analysis into the sport compared to other sports, thus, a higher chance the betting market is inefficient, meaning it can be beaten.

My approach, inspired by Nate Lateshaw of Literal Fight Nerd, was to create a hierarchical model structure. A hierarchical model consists of multiple layers, each building upon the outputs of the previous layer. The first layer model evaluates performances of fighter's broken down round by round, and outputs a stat called "expected round score", a predicted probability each fighter won a round in the eyes of judges, based off their individual performances in each round. The second layer model predicts the probability each fighter would win a fight, given information available prior to the fight, including the output from the first layer model.

[1]Steven D. Levitt, "Why are Gambling Markets Organised so Differently from Financial Markets?" The Economic Journal 114, no. 495 (2004): 223-246, https://doi.org/10.1111/j.1468-0297.2004.00207.x.

2 Methods

2.1 Data Collection

The data collection phase of the project involved a multi-stage process to gather relevant information from various sources. I used the python library beautiful soup to create web-scraping scripts to gather data from UFCStats.com and MMADecisions.com. The process involved scraping each website to gather the URLs for each UFC event, and subsequently each UFC fight. Lastly for each UFC fight URL, I scraped each respective web page for round level statistical data (e.g., significant strikes, control time, and kick accuracy), and judge scorecard information. This dataset includes 20 features from 34666 rounds. Additionally, I used a comprehensive UFC stats dataset from Kaggle[2] which included fight statistics and metadata from every UFC fight. This dataset has 48 features and 14130, 2 for each fight.

After cleaning and preprocessing the data, I merged the two datasets. For each round of each fight in the UFC dating back to 1994, I had data on stats such as length of round, significant strikes, control time, etc. I also scraped all judges' scorecards for each round. Since there are three judges, it is impossible for a round to end in a tie. Thus, using scorecard information I was able to compute who won each round. My first layer model was built from this dataset. It takes in the general fight or round information, and statistics from the round to predict a winner of each round.

2.2 Feature Engineering

Both models required feature engineering to properly set up the data into for modeling. Firstly, there were a significant number of rows with missing data. The trend was these fighters tended to have shorter careers in the UFC, generally because they were unsuccessful. The lack of information was an indicator to the model to predict against them. However, dropping all rows with null values was not a great idea because this would have left too little training data, thus, I created strategies to impute the missing data with reasonable estimates of the true values. For fighters with missing weight values, I used the average weight for their weight class. Since people's wingspans are very similar to their height, for missing

[2]`https://www.kaggle.com/datasets/danmcinerney/mma-differentials-and-elo`

height values, I used their reach, and vice versa for missing reach values.

Additionally, I created new features from the dataset which I thought would be especially informative. Some examples include significant strikes per minute, days since last fight, and finish percentage (% of fights where a fighter wins by KO or submission), and lastly my first layer model's output of expected round score.

In both models I created features which calculated the differential or ratio. An example would be height ratio, which would indicate for example fighter 1 is 12% taller than fighter 2. In some cases, this was more informative to the model than listing their separate statistics, in other cases, it hurt the model's performance.

Many features took in a fighter's previous fight data, however, fighters in their debut inherently do not have past data. To get around this problem, I imputed data using global averages of all fighters in their first fight. Thus, with the lack of previous data, I treated all debut fighters as an "average" fighter, not having any other data to base predictions on. A caveat to this strategy is often when fighters move up to the UFC, one of the premier MMA promoters, they have had a previous MMA career in other leagues. Sportsbooks or bettors have access to this information, while my model was limited to just stats and data from the UFC only.

Lastly, data manipulation was required to put data into the correct format for modeling. For both models the dataset had 2 rows for each round, one for each fighter. Each row needed to be matched up so it corresponded to the same round or fight. Additionally, for the 1st layer model predicting rounds, I dropped all fights which did not end in a decision, as these there are no publicly released scorecards for these fights. This exclusion may have introduced a bias in my first layer model because fights that ended in KO or submission were not included.

2.3 Modeling

The goal of building the 1st layer model was to establish a solid foundation for round success prediction. The idea behind this model is it is a better way to evaluate fighters in their past fights. Simply considering wins/losses may give too much weight to a luck factor or the element of randomness in mixed martial arts. This model aims to cut through the noise by quantifying how dominant (or not) fighters were in their past rounds. Fighter's that barely win each round and end up winning a fight should be distinguishable from fighters who dominantly win their rounds. This expected round score model would exhibit this gap, while the wins/losses would only say both fighters won.

After organizing the data into a format compatible with being inputted to train a model, I used the random forest and artificial neural network models. Initially, the model, was performing extremely well in predicting fighter1's winning rounds, but quite poorly for fighter2. I realized in the dataset I had scraped, fighter1 was the winner of each fight, thus, won the majority, if not all, of the rounds in the fight. This data leakage led to the model guessing fighter1 most of the time because fighter1 did in fact win most of the rounds. To fix this issue, I randomly selected half of all rows in my dataset and switched the labels of fighter1 and fighter2. This decreased the overall accuracy of my model but increased the recall and f1 score of the fighter2 class predictions.

```
Accuracy: 0.9300822561692127
              precision    recall  f1-score   support

           0       0.93      0.99      0.96      1418
           1       0.90      0.65      0.76       284

    accuracy                           0.93      1702
   macro avg       0.92      0.82      0.86      1702
weighted avg       0.93      0.93      0.93      1702
```

```
Accuracy: 0.8836662749706228

Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.89      0.88       839
           1       0.89      0.88      0.88       863

    accuracy                           0.88      1702
   macro avg       0.88      0.88      0.88      1702
weighted avg       0.88      0.88      0.88      1702
```

Figure 1: 1st layer model before and after changing fighter1 issue

The baseline I compared to was looking at which fighter had more successful significant strikes in each round. This baseline yielded 74.4% accuracy, while my final model yielded 88.3% accuracy. The reasoning behind this is because significant strikes are a good indicator of success in MMA. In general, as can be seen from the baseline percentage, the fighter to land more significant strikes typically incurs greater damage, leading to victory more often.

The next step was creating the second layer and final model. The goal of this model was to take in statistics that would be available before a fight such as win/loss history, significant strikes data, and fighter's expected round history (first layer model) and predict the fight winner. This model used a lot more metadata about each fight than the first layer model. Therefore, increased feature engineering was required, such as imputing previous fight information. For this task, I tried out a variety of models such as RF, KNN, logistic regression, SVM, and gradient boosting. In the end, the logistic regression model performed the best, yielding 61.0% accuracy. The baseline I compared against was how often the predicted favorite according to the betting odds won, 64%.

3 Results

The first layer model performed very strongly. With an accuracy above 88%, the expected round metric made for a good input feature into the second layer model. The two models had identical scores on the testing data.

Table 1: Performance Metrics of 1st Layer Models

Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**
RF	88.36%	0.88	0.88	0.88
ANN	88.36%	0.88	0.88	0.88

Unfortunately, the results for the second layer model were disappointing. The best performing model was a logistic regression with, an l2 regularizer, which had a 61% accuracy. According to a dataset of UFC betting odds , in betting, the predicted "favorite" wins 64%. Thus, model came up short of being more predictive than betting odds

Table 2: Performance Metrics of Machine Learning Models

Model Name	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Random Forest	59.1%	0.59	0.59	0.59
ANN	59.1%	0.59	0.59	0.59
Logistic Regression	**61.0%**	**0.60**	**0.60**	**0.60**
SVM	57.12%	0.57	0.49	0.53
Gradient Boosting	55.71%	0.54	0.55	0.55

A notable observation from the analysis was that the features weighted most heavily by the Logistic Regression model differed significantly from those prioritized by the Random Forest model. This discrepancy can be attributed to the inherent differences in how these models evaluate feature relationships. Logistic Regression emphasizes features based on their direct linear relationship with the outcome variable, assigning clear coefficients to each feature. In contrast, Random Forest assesses not only linear relationships but also more complex interactions between features, often leading to different features being highlighted as important.

Table 3: Feature Importance of Logistic Regression and Random Forest Models

Logistic Regression		Random Forest	
Feature	Coeff.	Feature	Coeff.
reach_fighter1	0.08	age_fighter2	0.06
reach_fighter2	-0.08	age_fighter1	0.06
sig_strike_per_min_f1	0.07	prev_elo_fighter1	0.04
height_fighter1	-0.06	L15_ER_avg_f1	0.04
height_fighter2	0.05	prev_elo_fighter2	0.04

4 Conclusion

While I believe the task of to using machine learning to predict UFC fights is feasable, there may be nuances the human eye can pick up but are not being captured in the statistics. Betting odds are initially based on bookmakers' best estimates, but through price discovery, and wisdom of the crowd, are eventually molded into the final price, which is what is being considered here as the baseline. Bettors can use both their models as well as their intuition and domain knowledge to inform their decisions in placing their bets, which ultimately leads to the odds moving towards efficiency. There is a disconnect between what this machine learning model is capturing vs. what the betting markets are considering to be important feature with high levels of prediction power.

Although the results of the model were not quite as high as I was hoping for, the skills developed during this project are extremely valuable to me and my passion for data science, specifically in sports. I feel more comfortable taking on big projects in the future, having gone through this process. After collecting, cleaning, manipulating, and analyzing data, I feel equipped to do it again, with a level of expertise which will come in handy when I inevitably run into similar problems as I have run into in this project. Addressing these problems and organizing the work in a way which would mitigate future problems from occurring is a valuable skill that I have developed through this experience, and it will undoubtedly serve me well in future related projects and endeavors.