UFC Fight Predictor

Mixed martial arts is one of the fastest growing sports in recent history and in 2019 it made a 5 year deal with ESPN to broadcast some of its fights. With any sport of this size people want to know who is gonna win before the fight even happens whether to brag about your knowledge of the sport or make a few safe bets for some extra cash.

1. **Data**

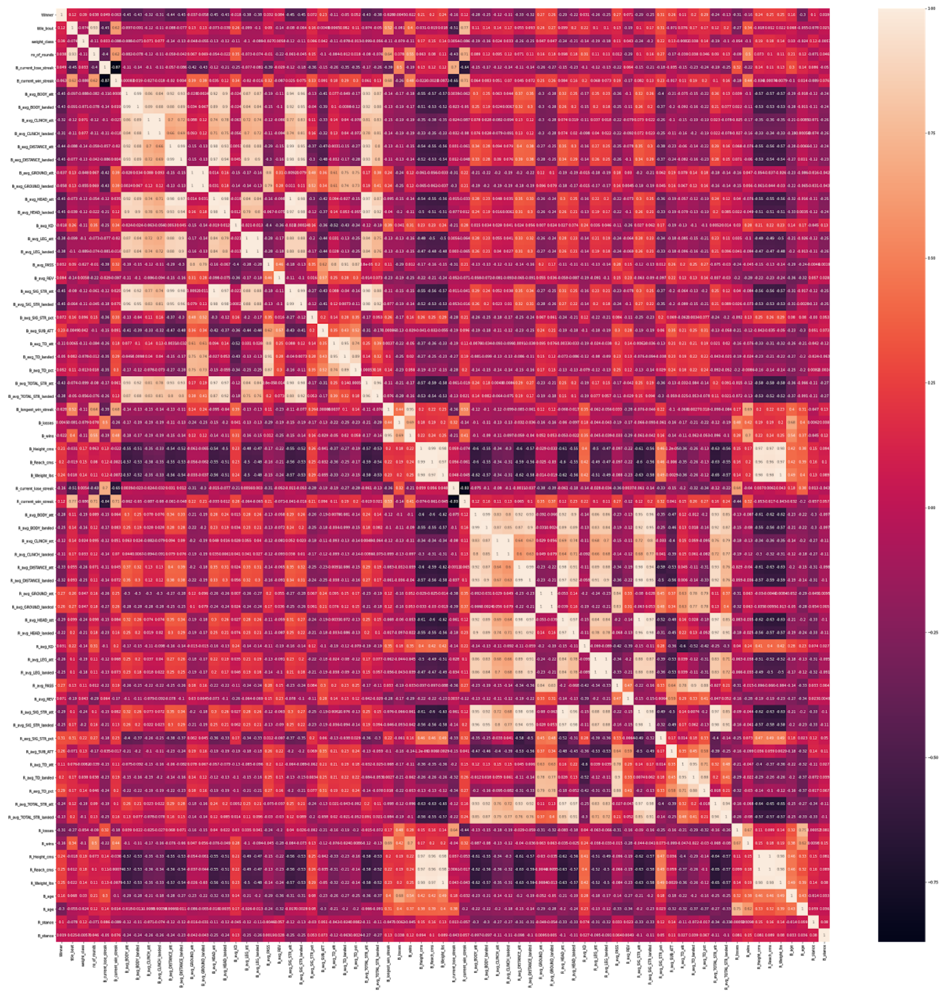
The data was all taken from a very well put together kaggle database. The dataset contained a list of every fight from the UFC's start in 1993 to june of 2019. This data contains all sorts of information about the fight where it took place, who was refereeing, and all sorts of fight statistics.

2. **Data Wrangling**

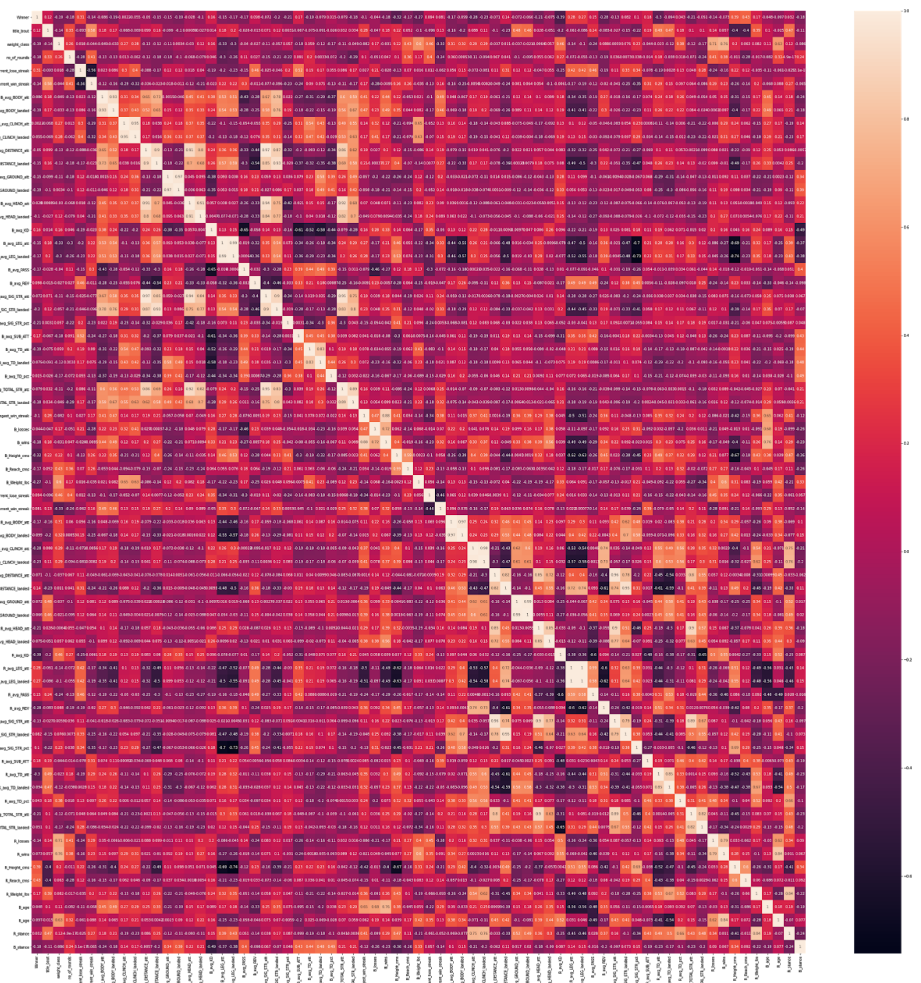
The cleaning for this data was very easy. This dataset is very well put together as I mentioned before but there were a few things I did need to do. The first was to drop all the fights that were before the rules for the UFC were made and used because those were not the same type of fights we see now those were basically anything can go. Then I dropped most of the useless columns there were such as referee, location, and date. I also made sure to drop all the draws that happened as I was going to make this a binary classification problem trying to predict if the red fighter won which would be a 1 and if blue won it would be a 0. Finally after seeing there were still nan values after all this I decided to drop all the rows with nan values since there was no good way to replace that data.

3. **EDA** [Visualizations]

All Fighters Heatmap



Cowboy Heatmap

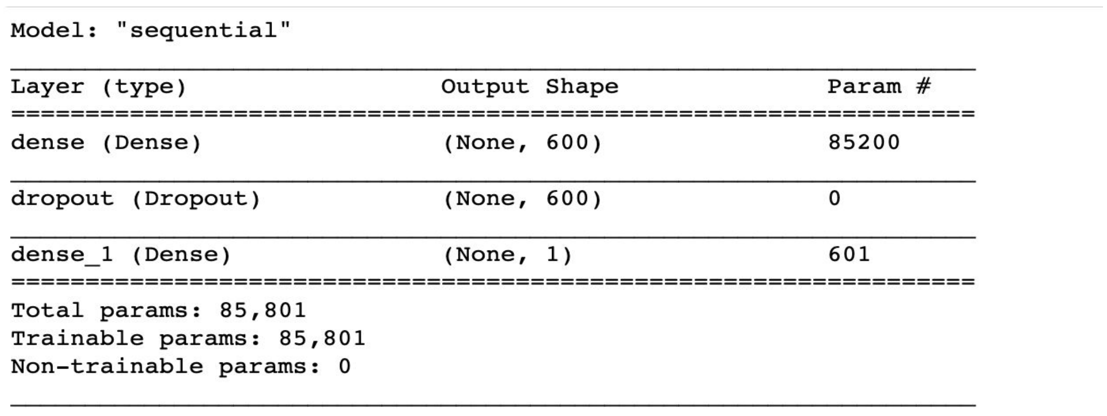


During my eda I had to look at all 140+ columns and decide how I wanted to use them or if I wanted to use them as that is a lot of features. There would be one specifying how many strikes were thrown another for how many landed it also had a wide variety of different strikes and positions whether they were thrown standing up, in the clinch, on the ground, or just specifically the head. There were also many columns talking about what their average opponents looked like I was not sure if I should use these columns as we were just trying to see if it could predict the fight at hand.

This is where I decided to use 2 different datasets one with all the features and another with less features trying to keep only the 70 or so I thought to be truly useful. This did not really make a difference to the model in the end but is an important note about this dataset. Also Something I found very interesting while looking at my eda I saw these 2 heatmaps one of all the fighters and the other of a specific fighter Donald Cerrone or also known as Cowboy. These 2 heatmaps are very clearly different from one another as we can see for most fighters the leg kick are a very unimportant feature but for cowboys heatmap we see that they are crucial for his style. This made me realize that I should possibly encode the fighter names since they are a factor as no fighter fights exactly like another and some can be completely. This is because mma is a very diverse sport with many different martial arts ranging from wrestling all the way to muay thai.

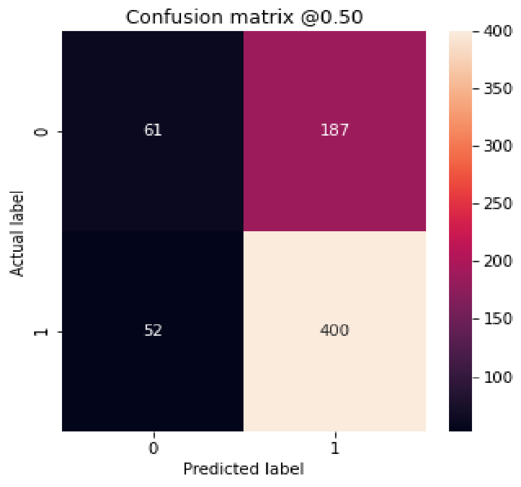
4. Deep Learning

For my deep learning models I used tensorflow to create my prediction model and I tried 2 different datasets. For the datasets one was 141 columns and the other was 72 columns. I found that the model performed a little worse with the 72 column one so I dropped it all together. Then I tested the dataset as a whole then with a balance of 50% red wins and 50% blue wins. As seen below by the metrics we see that the skewed dataset did perform a little better than the balanced dataset. For my models on the other hand I used different forms of weights for each model. My first was with just initial weights which I found performed better than than the model that had class weights attached to it. That said the class weight model did perform a lot better finding the true negatives than the other model but at a great cost of the true positives.

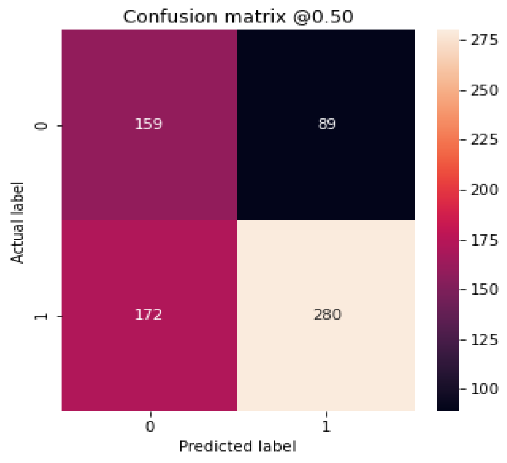


```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 600)               85200
_________________________________________________________________
dropout (Dropout)            (None, 600)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 601
=================================================================
Total params: 85,801
Trainable params: 85,801
Non-trainable params: 0
_________________________________________________________________
```

Models with Imbalance Data



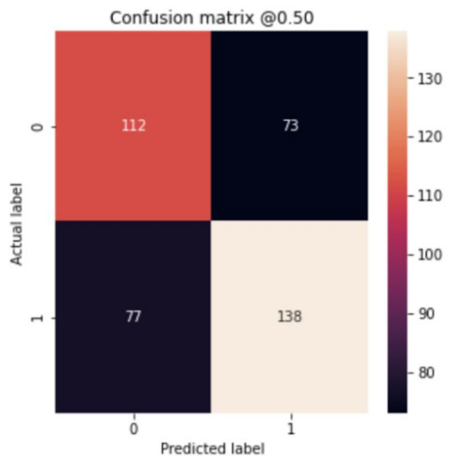
Initial weights only



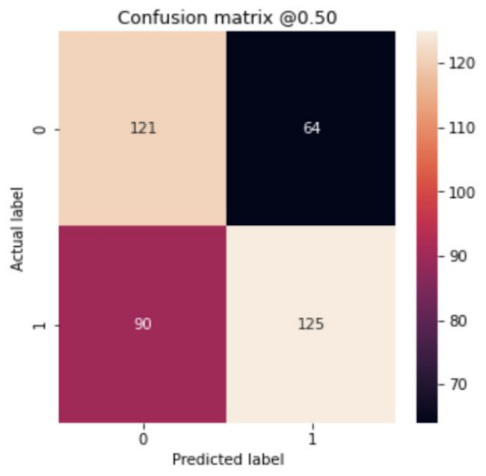
Class + Initial weights

loss : 0.60981565713	loss: 0.63559824228
accuracy : 0.65857142210	accuracy: 0.62714284658
precision : 0.6814309954	precision: 0.75880759954
recall : 0.88495576381	recall: 0.61946904659
auc : 0.68273621797	auc: 0.69237530231

Models with Balanced Data



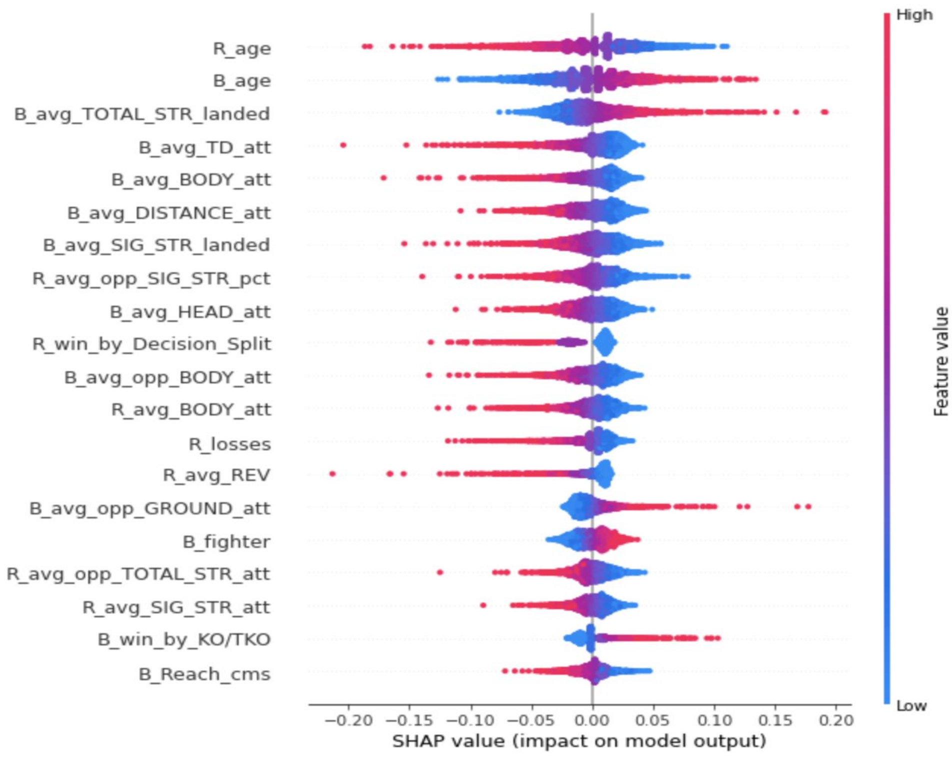
Initial Weights Only
loss : 0.71151006221
accuracy : 0.625
precision : 0.65402841567
recall : 0.6418604850
auc : 0.6682715415

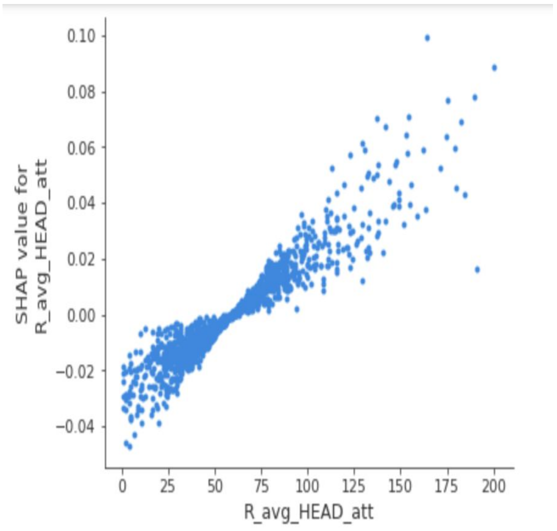


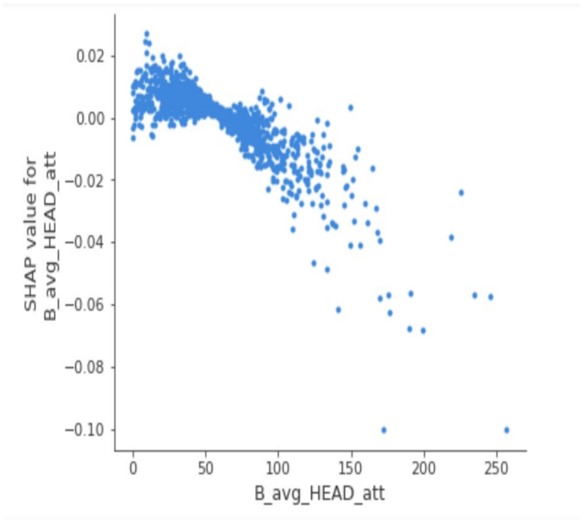
Class + initial weights
loss : 0.90262174606
accuracy : 0.61500000953
precision : 0.66137564182
recall : 0.58139532804
auc : 0.65586429834

5. Feature Importance/Conclusions   Extra Visualizations

While looking at the features it is important to remember our model was trying to predict whether the red fighter would win or not. If you click on the extra visualizations and look at the dependence plot you can see even more in detail on how each feature affected the model. A good example is the head attack features for each fighter that you can see below.As each graph shows you can clearly see since we are predicting the reds fighter winning that the more head strikes the red fighter had the greater impact it had on the model because the more time he hits the opponent in the head the more likely he will win that fight as head damage is no joke in any sport. The same goes for the blue fighters graph, remember that the model predicts red fighter wins so the more head strikes the blue fighter has the more damage he is doing to the red fighters head which would result in him potentially winning that is why you see a negative effect on the model the more head strikes the blue fighter has. This can be seen with many of the other features as well if a feature has a high number of strikes attacked by the red fighter it will affect the model positively and vice versa for the blue fighter features. After noticing this I realized my model must have a hard time predicting comebacks as any model would because the data will always be heavily skewed in the other direction since a comeback is usually a lucky or well placed strike that knocks their opponent out.







For some final conclusions on the model I would highly recommend any average UFC fan use this before making a bet. While it does favor tend to favor the red fighter it still does predict accurately almost 70% of the time which is a lot better than just taking your average guess and that can help someones betting gains a lot.

6. **Future Improvements**

Would really like to play around with different layers. I only chose these model metrics because they seemed to work consistently the best. Would like to see how adding some layers with a tanh activation function would change the model outcome. I would also like to try some more different datasets as well because they tend to be very different styles of fighting throughout the different divisions so maybe just using specific divisions when training would help with the prediction outcome at that level.

Thank you to Rahul Sagrolikar best mentor on this project I coulda asked for wouldn’t have been able to do it without him.