MMANET: Trying “MMA Math”

Final Report - ECE324

Rui Gao, Johnny Ye

Professor Jonathan Rose

Word count: 1974

Permission to post video: Yes
Permission to post final report: Yes
Permission to post source code: Yes

Introduction

Though there are in-depth analysis of advanced metrics in other sports, MMA is lacking in this aspect, and the application of machine learning is nonexistent. We propose a neural network to analyze the thousands of UFC matches, with the goal of predicting the winner using past fight data.We hope that by leveraging a neural network, we can glean insight about the factors determining the outcome of a match. Such fight statistics include strikes attempted, strike accuracy rate, cumulative win and loss record, and control time in the ring. Creating a predictor could have real impacts: coaches and fighters can analyze their chances against opponents and use the insights to prepare for their opponent, and sports gamblers can make more informed decisions in placing bets. Finally, the UFC can predict what fights are close or plain mismatches, and can plan matches accordingly.

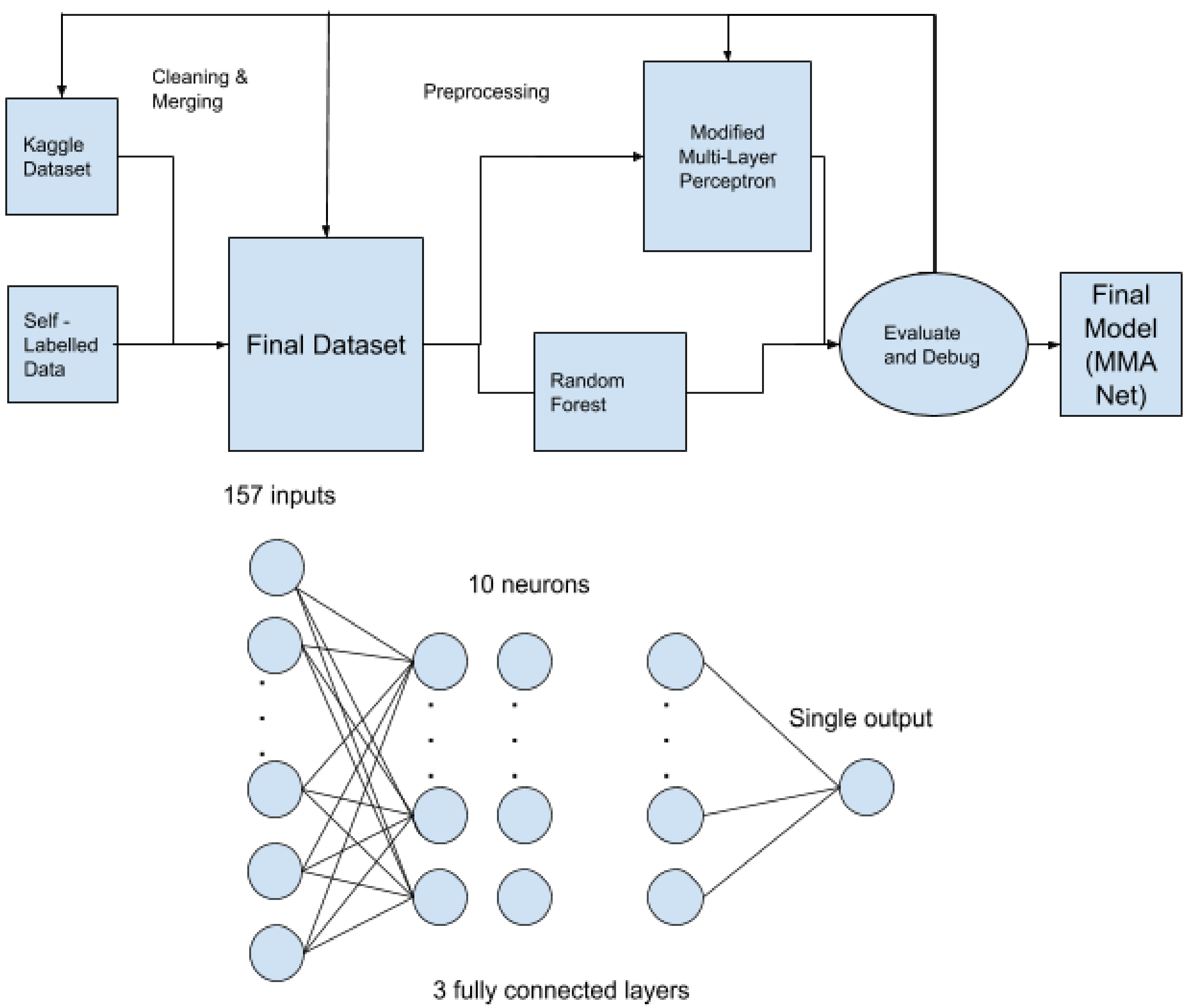

Figure 1: General project outline and model structure

Background

There have been a few personal projects to predict mma fights, yet none were made into papers or software products. One such project, done by Christopher Ho for the Stanford ML Course, used an MLP to predict mma outcomes, among other approaches such as SVM and Naive Bayes [1]. Though the accuracies only reached 67%, he only applied basic models, with no use of new techniques such as batch normalization and regularization. Even with the low rate of success, the MLP seemed to be the most promising approach [1]. Other personal projects exist, with other approaches such as random forests and simple rule-based heuristics. One approach, a personal website for MMA prediction using Vegas betting lines, managed to achieve a 71% accuracy [2]. However, we will not be using Vegas odds, as it is not related to the fighting factors is a human--made prediction.

More generally, sports prediction models exist for both commercial and research purposes. A generalized sports prediction model is given by Unanimous AI [3], however, not much info can be found on the techniques used. In terms of research, a paper from 2017 gives a rough survey of machine learning techniques used for sports prediction [4]. It shows that ANN's like the MLP we are using is a common technique although no past work exists in MMA.

Data

Collection

The source for our dataset can be found at: https://www.kaggle.com/rajeevw/ufcdata. The `preprocessed_data.csv` file contains a label for the winner and 13 columns labelling the weight class. The remaining 144 columns are statistics from the two fighters' past fights with 72 statistics for each fighter. The fighters' names are removed and changed to either red or blue. This gives a binary classification problem with 157 features. To analyze more granular data, we used the dataset found at: https://www.kaggle.com/calmdownkarm/ufcdataset. For this data, there were 435 fight statistics for each fighter, 8 fighter details for each fighter, 5 columns for event information such as date, maximum amount of rounds, and finally 2 columns for the winner and the win type. This dataset was used to see if more granular statistics would increase model accuracy.

Cleaning

The dataset needed to be cleaned, as some values had empty values. Empty values in fight statistics meant no prior history of the fighter, so this was filled with 0, but an empty value for the fighter details had to be removed. As well, the classes were imbalanced as shown below.

```
Red     2380
Blue    1212
Name: Winner, dtype: int64
```

Figure 2: Winner value counts

The name was firstly removed, and were referred to as Red and Blue arbitrarily depending on the placement of their fighter details. Since these labels are arbitrary, we can balance it by simply rearranging the placement of Red and Blue fighter details.

Certain columns were dropped, such as the “title_bout”, as this information is encoded in the number of rounds column, “no_of_rounds”.

Winner	title_bout	no_of_rou	B_current_	B_current_	B_draw	B_avg_BO	B_avg_BO	B_avg_CLI	B_avg_CLI
Red	TRUE	5	0	4	0	9.2	6	0.2	0
Red	TRUE	5	0	3	0	14.6	9.1	11.8	7.3
Red	FALSE	3	0	3	0	15.35484	11.32258	6.741935	4.387097
Blue	FALSE	3	0	4	0	17	14	13.75	11
Blue	FALSE	3	0	1	0	17	14.5	2.5	2
Red	FALSE	3	0	4	0	19.5	12.33333	11.83333	7.166667
Red	FALSE	3	0	3	0	15	7.416667	6.083333	3.416667
Blue	FALSE	3	1	0	0	16.25	11	17.5	11.25
Blue	FALSE	3	0	1	0	7.25	4.75	1.75	0.5
Red	FALSE	3	0	1	0	25.4	17.9	22.5	16.8
Blue	FALSE	3	1	0	0	5.428571	4.142857	12.28571	9.142857
Red	FALSE	3	0	2	0	21.875	13	14.875	10.25

Figure 3: Uncleaned and unbalanced data

Winner	no_of_rou	B_current_	B_current_	B_draw	B_avg_BO	B_avg_BO	B_avg_CLI	B_avg_CLI
0	3	1	0	0	1	0	0	0
1	3	1	0	0	6.5	2.666667	8.833333	5.166667
1	3	0	1	0	38	31	87	68
0	3	0	4	0	11.25	9	13.5	7.5
0	3	1	0	0	4.125	2	5.875	2
1	5	0	1	0	11	9	2	2
0	3	2	0	0	2.5	2	2	1.5
0	3	1	0	0	16.08333	9.5	1.833333	1.166667
1	3	0	2	0	6.333333	3.666667	14.33333	7.666667
1	3	0	3	0	1	0.666667	9.666667	7.666667
1	5	0	2	0	4.25	3.875	5.375	4.125
0	3	1	0	0	0.5	0	2	0

Figure 4: Cleaned and balanced data.

Visualization

We created some violin plots to see how well features split the data for our classification problem. Some of the plots that are split the best are shown below: no large splits exist but we do see weak correlation.

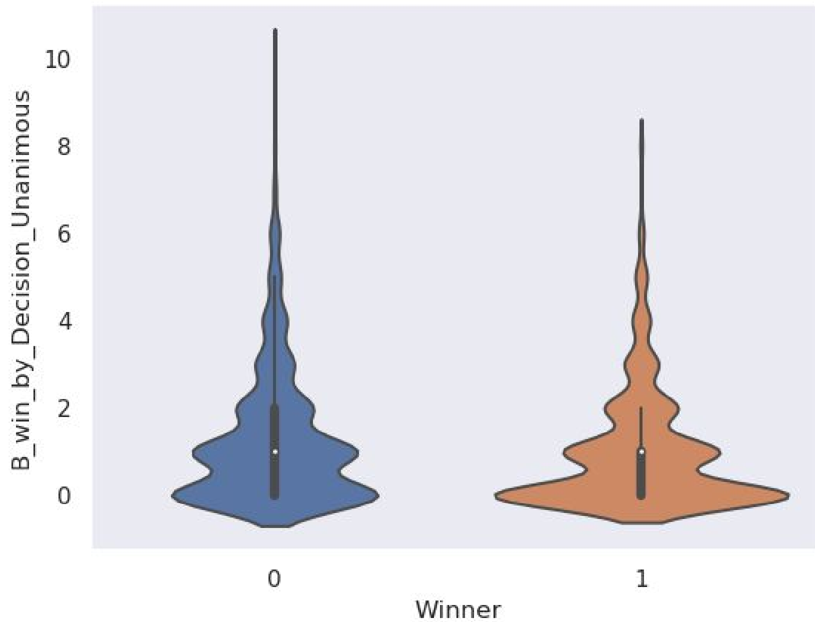

Figure 5: Wins by unanimous decision for blue(0) vs winner. Shows a correlation between past unanimous wins and the Winner.

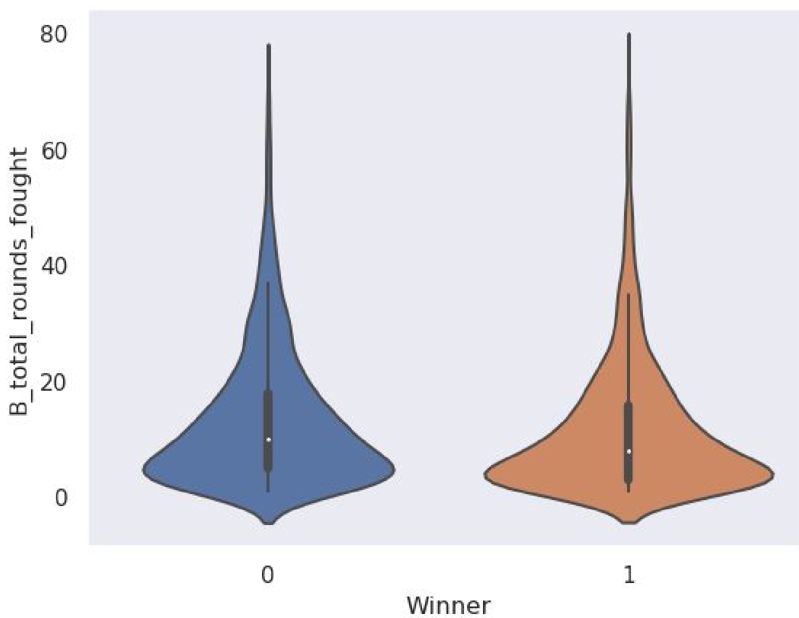

Figure 6: Total rounds fought by blue(0) vs winner. We see the blue plot is slightly higher. Shows a correlation between experience and winning.

Preprocessing

We chose an 80/10/10 split for training validation and testing respectively. We shuffled the data randomly prior to the split and we normalize each column so that values are between 0 and 1. We also have datasets normalized to zero-mean and unit variance, but they were not used as they dropped baseline testing accuracy by ~7%.

Baseline

We have chosen to use sci-kitlearn's Random Forest classifier with 100 decision trees and no maximum depth for the trees. The random forest model then uses bootstrap aggregation to decrease variance/overfitting and make the model generalize better.

Architecture

The best model architecture so far is an MLP with 3 hidden layers, all with 10 neurons each. The input layer takes in an input of 157 features, and the output layer has one neuron with a binary classification. The sigmoid function was used on all layers, and batch normalization was applied after all layers except for the last. The model has only 1871 parameters to train.

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Linear-1               [-1, 10, 10]           1,580
       BatchNorm1d-2               [-1, 10, 10]              20
            Linear-3               [-1, 10, 10]             110
       BatchNorm1d-4               [-1, 10, 10]              20
            Linear-5               [-1, 10, 10]             110
       BatchNorm1d-6               [-1, 10, 10]              20
            Linear-7                [-1, 10, 1]              11
================================================================
Total params: 1,871
Trainable params: 1,871
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.01
Forward/backward pass size (MB): 0.00
Params size (MB): 0.01
Estimated Total Size (MB): 0.02
----------------------------------------------------------------
```

Figure 7: torchsummary output

Loss Function: BCELoss
Optimizer: Adam, (weight decay of 0.001)
Batch size: 64
Learning rate: 0.0001

Feature Extraction

In an effort to reduce the dimensionality of the data to improve the performance of our model, we experimented with several feature extraction techniques: Principal Component Analysis, Independent Component Analysis, Locally Linear Embeddings, t-distributed Stochastic Neighbor Embedding and Restricted Boltzmann Machines. We applied the feature extraction to the full dataset and then experimented by training our baseline with the new features.

Some graphs of different techniques reducing the data to 2 dimensions is shown below:

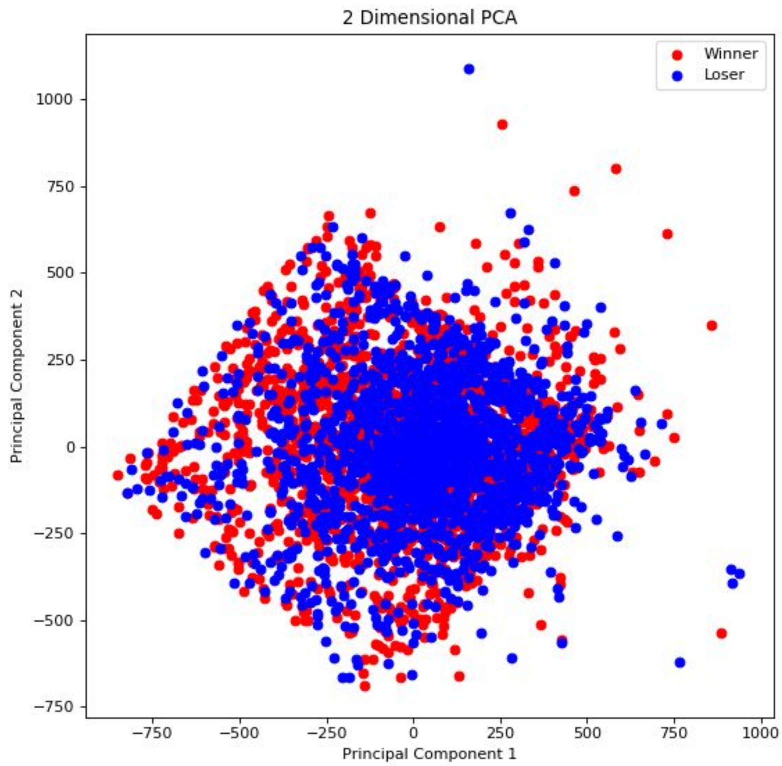

Figure 7: Principal Component Analysis to 2 Dimensions

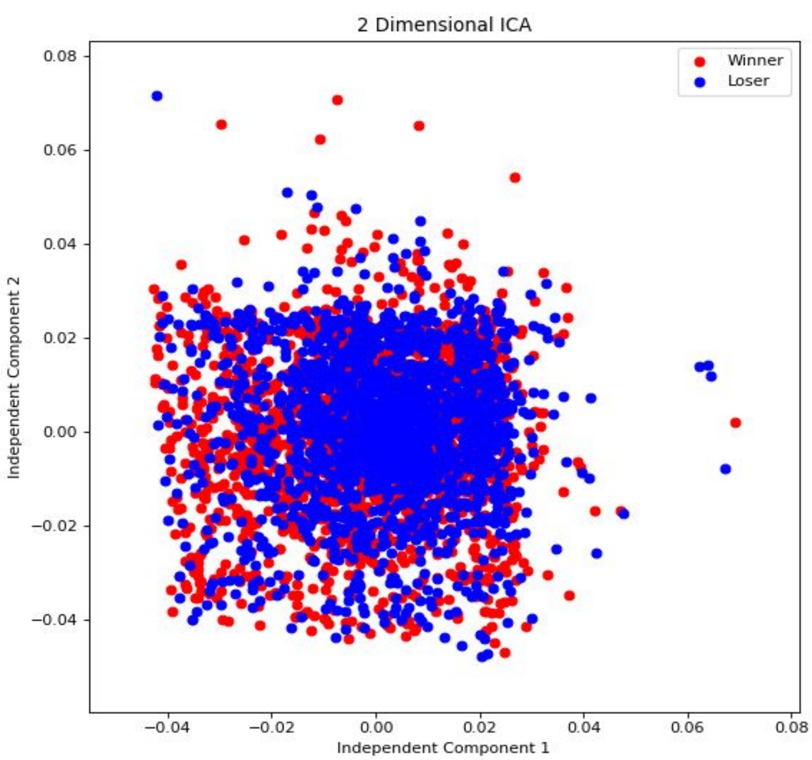

Figure 9: Independent Component Analysis to 2 Dimensions

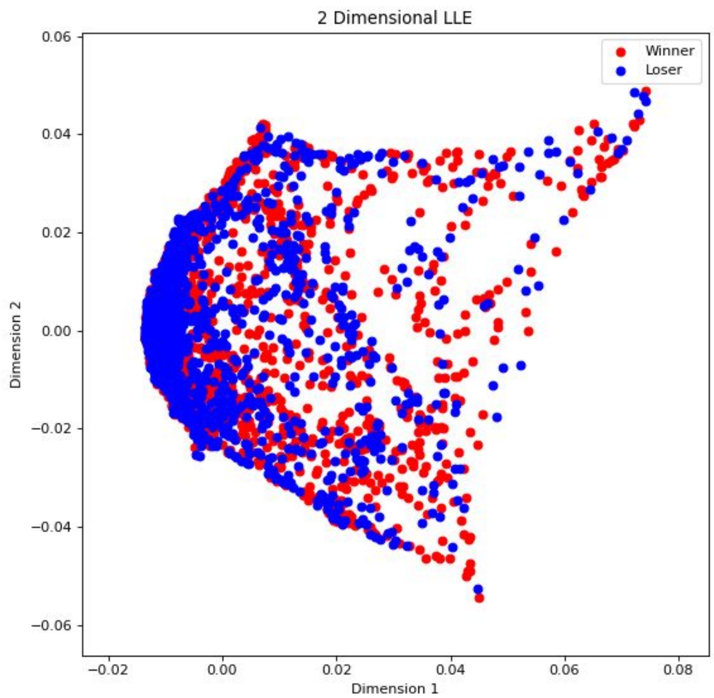

Figure 10: Locally Linear Embeddings to 2 Dimensions

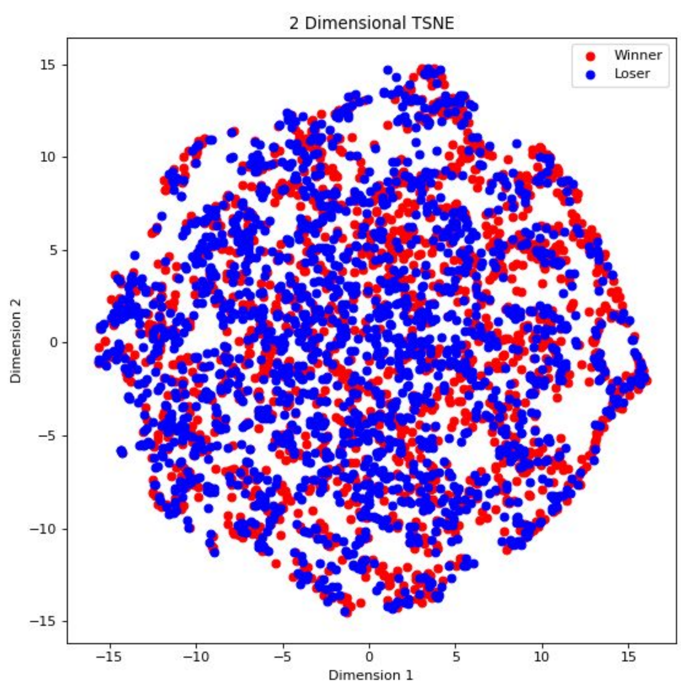

Figure 11: t-distributed Stochastic Neighbour Embeddings to 2 Dimensions

To help visualize how the baseline would work with the 2-Dimensional data, we show the decision boundary in Figure 12 below overfitting on the features from the Principal Component Analysis:

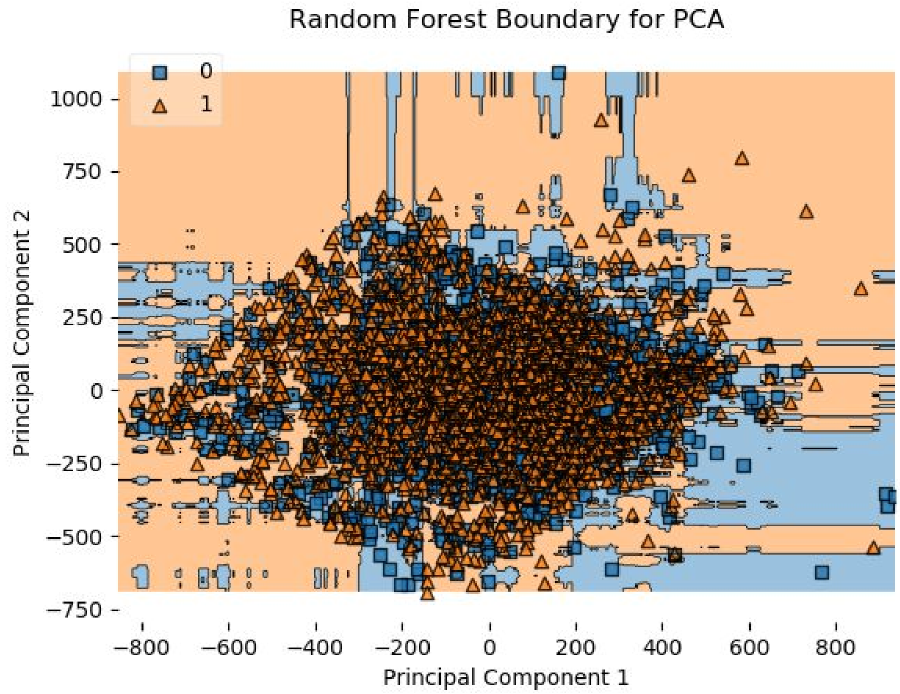

Figure 12: Decision Boundary for baseline when overfit on the 2-dimensional PCA data

As we see that the class is not separated after feature extraction and it is difficult for the baseline to draw a good decision boundary, we do not expect the feature extraction to improve model performance when testing on the baseline. Nonetheless, we experiment with reduction to different dimensions and we check for improvement. The results are shown below, all feature extraction is done with default sklearn settings:

Feature Extraction Type	Number of Dimensions	Validation Accuracy
PCA	12	54.44%
PCA	64	49.44%
ICA	12	50%
ICA	64	49.17%
LLE	12	50.83%
LLE	64	45.56%
t-SNE	2	50.83%
t-SNE	3	51.39%
Restricted Boltzmann Machine	N/A	53.06%

Table 1: Feature Extraction Results

Feature extraction does not increase performance so we do not use it. However, from this analysis, we can see the complications in working with our data from the difficulty of getting a good split between the classes.

Results

The baseline performs well, with hyperparameters tuned to get validation accuracy of 65.8% and then evaluated on the testing set to get 65% testing accuracy. Using 5-fold cross validation we get an accuracy of 61.25%.

The MLP is returning a maximum validation accuracy of 67.5% with a corresponding test accuracy of 65.8%. Below are the relevant accuracy and loss graphs for both training and validation.

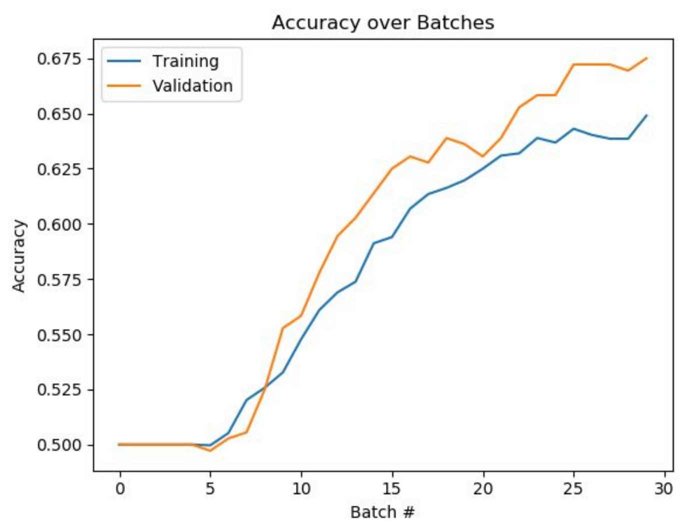

Figure 13: Training and validation accuracy for best model

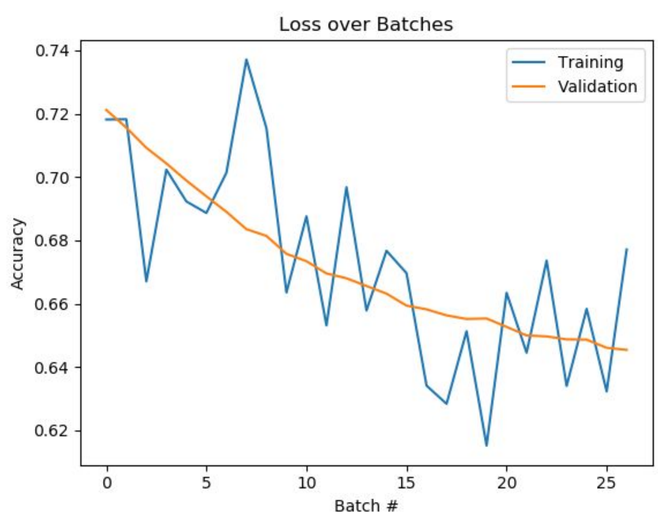

Figure 14: Training and validation loss for best model

```
Validation dataset accuracy: 0.661
Validation dataset accuracy: 0.653
Validation dataset accuracy: 0.675
Validation dataset accuracy: 0.672
Test dataset accuracy: 0.658
```

Figure 15: Validation and testing accuracy for best model

After achieving this accuracy, we attempted to use the more granular dataset mentioned above. We attempted the same model with further tuning, but we only achieved validation and testing accuracies of 55% and 54% respectively. This was done to see if granular data would have improved the model, but judging by the reduction of accuracy we decided the original dataset was adequate.

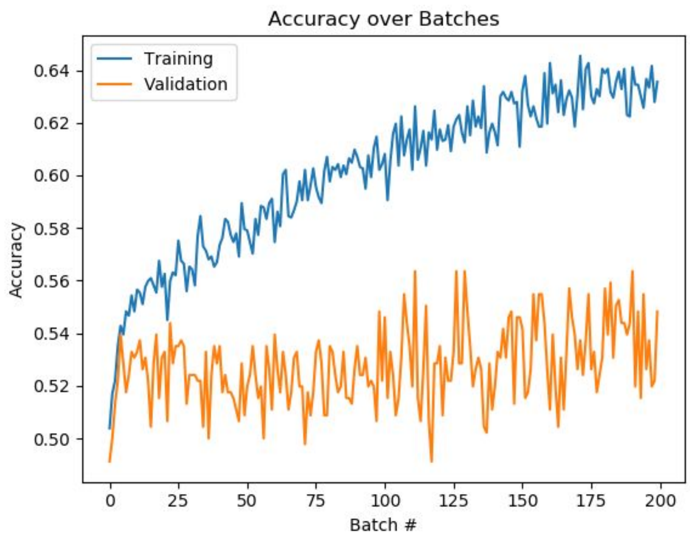

Figure 16: Training and validation accuracy for second dataset

As a final check that the model wasn't overfitting on the dataset, we implemented three-fold cross validation with the skorch library, using our best model. Since we could not implement a BCELoss function, we simply used the cross-validation as a test to see whether the model was overfitting, which would be indicated by large accuracy disparities across the different folds. However, as seen below, our model returned roughly the same accuracies for the three folds.

epoch	train_loss	valid_acc	valid_loss	dur
97	0.0385	0.5750	0.0383	0.0928
98	0.0381	0.5771	0.0379	0.0967
99	0.0377	0.5792	0.0375	0.0918
100	0.0373	0.5771	0.0371	0.1037

97	0.0386	0.5729	0.0384	0.0878
98	0.0382	0.5729	0.0380	0.0918
99	0.0378	0.5708	0.0376	0.0987
100	0.0374	0.5708	0.0372	0.0988

97	0.0385	0.5771	0.0386	0.1007
98	0.0381	0.5771	0.0382	0.1137
99	0.0377	0.5729	0.0378	0.1027
100	0.0373	0.5729	0.0374	0.1107

Figure 17: The final validation accuracy for each of the three folds

After these checks, we then tested the model on recent fights weren't included in the datasets. Vegas betting odds were collected from www.bestfightodds.com and compared to the model predictions.

Red Fighter	Blue Fighter	Winner	Vegas Prediction	Model Prediction
Nate Diaz	Jorge Masvidal	Blue	Blue	Blue
Kelvin Gastelum	Darren Till	Blue	Red	Red
Stephen Thompson	Vicente Luque	Red	Red	Blue
Derrick Lewis	Blagoy Ivanov	Red	Red	Red
Kevin Lee	Gregor Gillespie	Red	Blue	Red
Johnny Walker	Corey Anderson	Blue	Red	Blue
Makwan Amirkhani	Shane Burgos	Blue	Blue	Blue
Brad Tavares	Edmen Shahbazyan	Blue	Blue	Red
Katlyn Chookagian	Jennifer Maia	Red	Red	Blue
Julio Arce	Hakeem Dawodu	Blue	Blue	Blue
Robert Whittaker	Israel Adesanya	Blue	Red	Blue
Al Iaquinta	Dan Hooker	Blue	Blue	Blue
Luke Jumeau	Dhiego Lima	Blue	Blue	Blue
Ji Yeon Kim	Nadia Kassem	Red	Red	Blue
Jan Blachowicz	Jacare Souza	Red	Red	Red
Charles Oliveira	Jared Gordon	Red	Red	Blue
James Krause	Sergio Moraes	Blue	Red	Blue
Francisco Trinaldo	Bobby Green	Red	Red	Blue
Warlley Alves	Randy Brown	Blue	Red	Blue
Douglas Silva de Andrade	Renan Barao	Red	Red	Blue
Greg Hardy	Alexander Volkov	Blue	Blue	Blue
Calvin Kattar	Zabit Magomedsharipov	Blue	Blue	Blue
Ramazan Emeev	Anthony Rocco Martin	Blue	Blue	Blue
Stevie Ray	Michael Johnson	Red	Red	Blue
Beneil Dariush	Frank Camacho	Red	Red	Red
Chris Weidman	Dominick Reyes	Blue	Blue	Red
Jeremy Stephens	Yair Rodriguez	Blue	Blue	Blue
Manny Bermudez	Charles Rosa	Blue	Red	Blue
		Accuracy:	**0.75**	**0.642857143**

Figure 18: 28 recent fights, with vegas predictions as well as the model predictions.

Discussion

The performance of the baseline makes predictions at roughly the same accuracy as the MLP even after both have had their hyperparameters tuned. The model is doing well, but not much better than the baseline. Due to the rather small dataset we have, the MLP does not train as well. The model is quick to train on account of the small size, but a larger model would overfit within 20 epochs. Early stoppage returned the best possible validation, though we can see that validation accuracy is higher than training. This is most likely due to the two datasets having slightly different statistics due to the small size, and resulting in a different model performance for the different datasets.

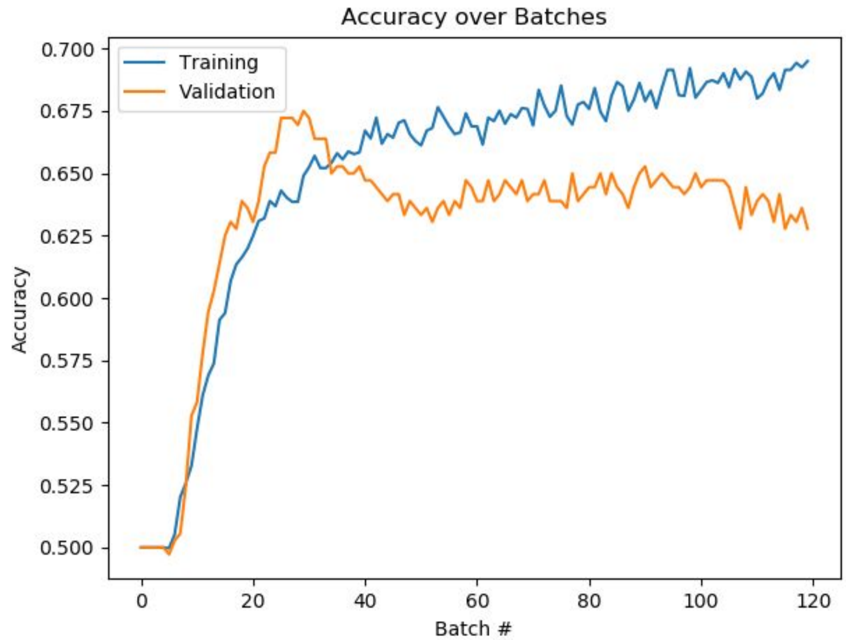

Figure 19: Overfit training

From the small sample we chose to compare with Vegas betting odds, we see that Vegas has a higher success rate, with an accuracy of 75% compared with our model prediction of 64%. Although disappointing, it is understandable as many factors affecting the fight cannot be included into the data, such as the emotional state of the fighter, hometown bias, quality of the training camp and the gym etc. This is not surprising, as we see that both quantitative and qualitative factors behind a fight must be considered for the best possible prediction.

A key insight we took from this project was the importance of having enough data. We chose to only use UFC data, as the UFC is the premier organization and had a virtual monopoly on elite fighters, and we wanted to use only the most high-quality fights for training. This resulted in only roughly 3500 fights. A simple comparison is most illustrative: most NBA players will log tens of thousands of game minutes, through many years and hundreds of games. In contrast, MMA fighters almost never have more than 30 professional fights, with each fight being maximum 25 minutes. There are also less than 10 fights every week from the UFC.All these factors lead to a general lack of data to work with. As a neural network improves with large quantities of data, we find the lack of data to be the foremost reason for equivalent performance of the baseline and the MLP.

Another insight is that granular statistics didn't always work: when dealing with more features the model didn't gain any deeper insight than less features. Having more features also forced an increase in model size, making a model that could not avoid overfitting.

Learnings

For future projects with less of a time constraint, here are some next steps:

1. Video/image data

Images/Videos of the fighter could be used as input to a neural network. Although this would be a massive step in computational power, the insights gleaned from image data may prove useful in analyzing athleticism, quality of muscles, fighting style and other factors.

2. Weighing past wins and losses by strength of schedule

Though the current dataset considers the average statistics of previous opponents, an improvement could be to have separate sections for opponents beaten, and opponents lost to. As well, the average could instead be a weighted average, weighted higher for more recent wins/losses. This has the potential of being more relevant to a fighter's current skill-set, as recent wins and losses are much more telling than wins/losses many fights ago.

For one final note, Vegas odds are not incredibly accurate, with a long-term prediction accuracy of around 70%. This points to evidence that there truly is unpredictability within MMA fights, and it may be for a long time until enough data and a powerful model points otherwise.

Ethical Framework

All fighter data is public and the collection is of no ethical concern.

This model can be used for good, allowing fighters to understand what they should change or style changes that work against their specific opponent. Allows coaches to further understand the nuances that go into the fight outcomes. However, if the model is only given to specific fighters, this could cause imbalances and unfairness that would lead to injustice.

One important aspect is mismatches: sometimes much better fighters are scheduled against much worse fighters, leading to unnecessary harm and damage. This could be prevented by the model, which can predict the percentage likeliness of winning. This increases non-maleficence.

References

[1]C. Ho, “Does MMA Math Work?”. [Online]. Available: http://cs229.stanford.edu/proj2013/Ho-DoesMMAMathWorkAStudyonSportsPredictionAppliedtoMixedMartialArts.pdf. [Accessed October 24, 2019].

[2] “UFC MMA Predictor”. [Online]. Available: https://ufcmmapredictor.herokuapp.com/. [Accessed October 23, 2019]

[3] “Sportspicker AI”. [Online]. Available: https://unanimous.ai/sports. [Accessed October 23, 2019]

[4] R. Bunker and F. Tabhtah, “A machine learning framework for sport result prediction”, *Applied Computing and Informatics*, vol. 15, no. 1, p. 27-33, January 2019. Available: https://www.sciencedirect.com/science/article/pii/S2210832717301485. [Accessed October 23, 2019]