NFL BetaLine: Leveraging GRUs on Advanced NFL Metrics for +EV Betting Framework

Andrew Annestrand
University of Washington
atannes@uw.edu

Abstract

In 2018, the supreme court case *Murphy v. National Collegiate Athletic Association* ruled that the federal ban on state regulation of sports betting was unlawful. Since that ruling, sports betting has exploded into a multi-billion dollar industry. In this project, I set out to explore possible market inefficiencies by leveraging modern neural network architectures.

1 Introduction

This project first began as a survey on the power of neural networks when applied to predictions for the National Football League (NFL). The structure of the dataset that I used was historical games and their outcomes dating back to 2006. Reluctant to use often misleading box score statistics, I decided to include advanced premium statistics from a few NFL analytics companies. I explored many different ideas ranging from generative modeling of a team's season, to predicting scores of games, and ending up on predicting solely the outcome of a game (winner/loser) as my final model. I take advantage of a neural networks ability to output pseudo-probabilities (they are not explicitly probabilities, rather inferred) for classification tasks to develop a betting framework that yields positive expected value on bet returns. Before getting further into methodology and predictions, it's important to understand the dynamics of sports betting and moneylines.

1.1 Understanding Moneylines

Sportsbooks are large conglomerates that essentially function as a casino. Instead of taking bets from bettors in the form of card games, slots, or dice games, sportsbooks take bets on the outcome of sports games. Some of the more well-known sports bets include horse-racing, greyhound racing, and perhaps Soccer matches. This project focuses on the betting of American Football games and their outcomes (moneylines). Although they seem daunting at first, moneylines are simply just bets that include the odds associated with a team winning a game in a match up. Take the following example, the New England Patriots are playing the Dallas Cowboys at home.

Figure 1: An example of the two money lines in a proposition.

When assessing a moneyline bet the first thing a bettor must do is to see if a team is favored (indicated by the -) or considered the underdog (indicated by the +). Once the favorite has been identified, then we can understand the odds of the bets. In the above example, the Patriots favored at 225 simply translates to: the bettor must place down $225 at stake to win $100 (if the Patriots win the matchup). Vice versa, for the Cowboys +225 translates to: the bettor must stake $100 to win $225 (if the Cowboys win the matchup). If you bet on one team and the other loses, you lose your money. Based on these odds, we can calculate an implied probability for either of the outcomes (see 2).

Patriots Implied Probability = 225/(100+225) = .6923
Cowboys Implied Probability = 100/(100+225) = .3077

Figure 2: The implied probabilities based on moneyline odds.

Now the reader has all the information necessary to understand the betting framework going forward.

1.2 Dataset

As mentioned before, the structure of the dataset consists of most NFL games played since 2006. Each row in the dataset includes all necessary information about the game and its outcomes. After many hours of arduous downloading and data wrangling, I was also able to include some advanced football metrics for both team and player performance. Player production grades from Pro Football Focus (PFF) were included for both home and away teams. These grades are created by humans who watch hours of film and assign grades to players based on their performance. The reason why these PFF grades are important is that they have context embedded in them which is something that box score statistics fall short of doing. Only player grades from the most valuable positions on a team were included (Quarterbacks, Wide Receivers, Coverage Players, Pass Rushers). The other advanced metric I used was Football Outsiders Value of Average (VOA) which is agnostic to game outcomes and focuses on a team's performance compared to league average. Further, weighted versions of VOA metrics were used which gives more value to recent games over early season performances.

Game Metadata
- Home team, away team
- Temperature, wind
- Stadium type (dome, outdoors)
- Home team rest days, away team rest days

Game Betting Data
- Home moneyline, away moneyline

Game Outcomes
- Home score, Away score

***Pro Football Focus (PFF) Player Ratings**
- Individual player production ratings (only top WAR positions included)

***Football Outsiders Value Over Average (VOA)**
- Weighted home/away team special teams VOA
- Weighted home/away team offense VOA
- Weighted home/away team defense VOA

Figure 3: Features of each data sample.

An important note is that both of these advanced metrics are based on the performance of the team/player up to the current week of the game in a particular season. There is no cross-season information in our dataset. For this reason, the games in Week 1 of each season were dropped from the dataset.

2 Methods for Game Prediction

First approaches to game prediction involved fully-connected classifiers that only took in the current matchups data in order to make a prediction. These models performed decently with accuracies around 58%, however I was not satisfied. Instead of settling on this, I decided to train a model that could account for the games/stats of previous weeks. The rationale behind this is that teams can go on ”runs” of good performance, by including previous weeks hopefully the model can capture this. In order to account for this new time-series data, I switched to a recurrent neural network architecture.

2.1 Model Architecture

The model architecture was quite simple in terms of layer complexity. Since our dataset is relatively small compared to others in the deep learning landscape, it didn't make sense to build deeper models.In fact, attempts to do so would typically lead to unstable results. Below is the final models architecture:

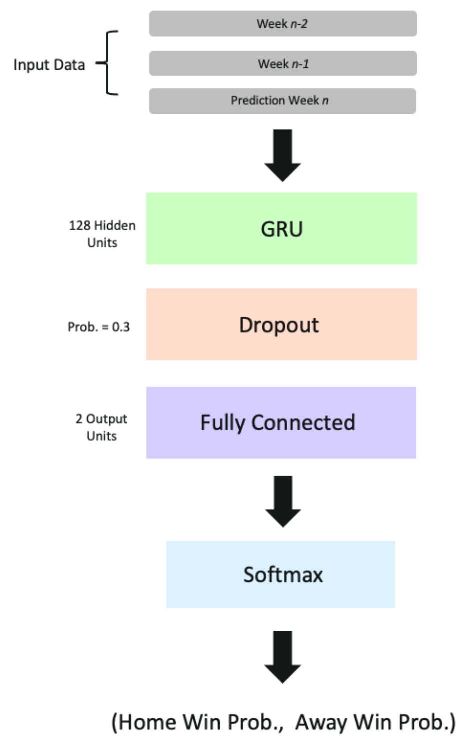

Figure 4: Neural Network Architecture.

The reader may wonder why GRUs were selected instead of other recurrent network layers. GRUs were chosen purely on empirical performance when training these models. LSTMs ran into problems with divergence a lot more often due to over-paramaterization. Because our setup had only short length sequences the use of LSTMs didn't really make sense (they excel and long-term memory).

2.2 Model Results

When evaluating the the training and validation results, the model was trained on 50 different randomly shuffled datasets. This was performed to get an accurate assessment of the expected performance over the whole dataset since its size is limited. There is a lot of variance in the sports games – some times we can get lucky, other times not. Thus, finding the expectation of our results is quite important.

The model ends up converging around the accuracy of 0.625 for validation test. Clearly there's some slight overfitting when comparing the performance of the model on the training vs. validation

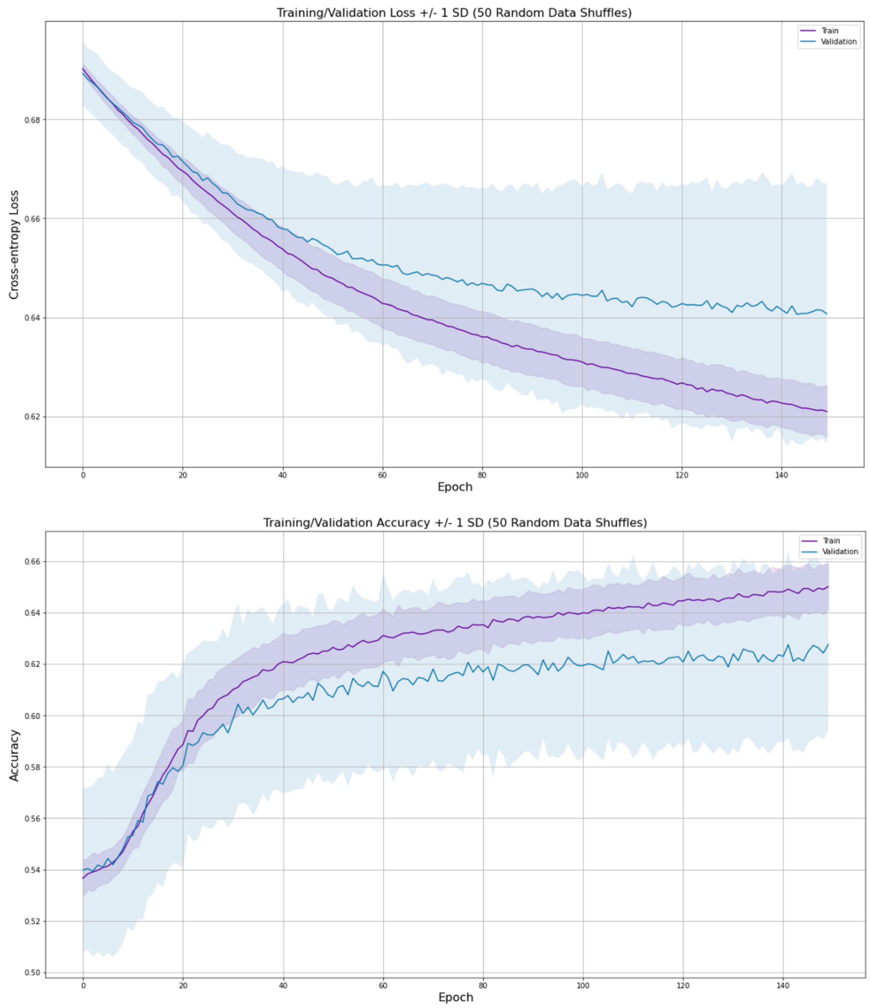

Figure 5: Loss Accuracy.

dataset. The dropout that was included in the model helped curb a lot of the overfitting that would happen in prior models. Overall this accuracy is quite nice considering noisy and variant the results of sports match ups can be. However, just this accuracy alone is not enough to help us make money betting. Applying the softmax on the outputs of neural network will give us a pseudo-probability that will allow us to develop our betting framework.

Table 1:

Results after 50 Different Trainings of 150 Epochs		
	Cross-entropy Loss (+/- 1 SD)	Accuracy (+/- 1 SD)
Train	0.6224 $\pm$.0056	.652 $\pm$.098
Validation	0.641 $\pm$.0291	.627 $\pm$.033
Test	0.646 $\pm$.019	.61953 $\pm$.031

3 Betting Framework

Hopefully the reader understands now that for each moneyline that is assigned to a team, we can retrieve an implied probability from it. The main crux of our betting framework is to expose discrepancies between the book's implied probabilities and our model's probabilities. One may wonder why there would be discrepancies in the first place if both are founded in expert knowledge/statistics. There is many reasons, but one that I believe to be most probable is that sometimes the market responds too strongly to noise. Humans love narratives and stories and often times bettors can get wrapped up in them leading to large bias. Alas, this framework takes advantage of the markets deviation from quantitative signals. A simple flow chart below shows the betting framework:

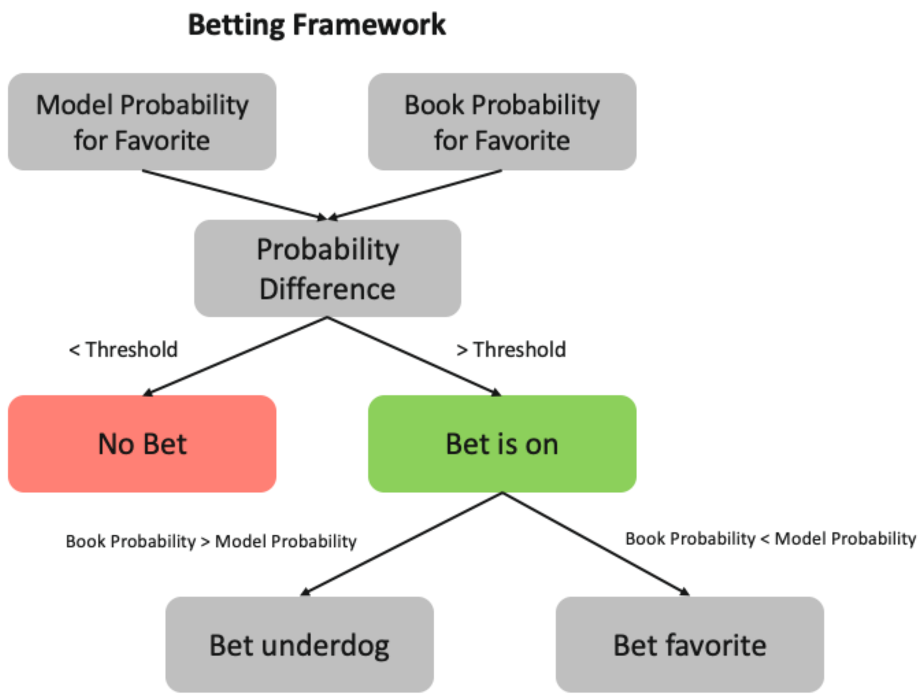

Figure 6: Betting Framework flowchart.

We evaluate a range of different thresholds for the probability difference used in the framework. If the difference is greater than the threshold, then we decide to make a bet on the game (we believe there is an edge to be had). The team we bet on is determined by the direction of the difference. Using the example from earlier in the report, lets say the sportsbooks have the Patriots favored at -225 and we are using a threshold of 0.03. This means the book has the Patriots winning the game at a probability of 0.6923. Now, we run this through our model and get a probability of 0.75 for the Patriots winning. Clearly the difference between the two surpasses our threshold so we will make the bet. Since our model is more in favor of the Patriots, we place the bet on the Patriots winning. Instead, if our model had the patriots winning at 0.55, then we would bet the Cowboys.

This exact framework was evaluated on 50 randomly shuffled runs and the results were pretty strong. All of the thresholds that we evaluated were at least positive expected value per bet. For the smaller threshold values, the expected value was mostly lower. This is likely to do with the framework simply triggering more bets at small edges thus raising the risk of losing. The zone that I consider the sweet spot is the 0.1-0.2 threshold range where where the distribution is much more narrow on the positive EV side while still triggering bets roughly 25% of the time.

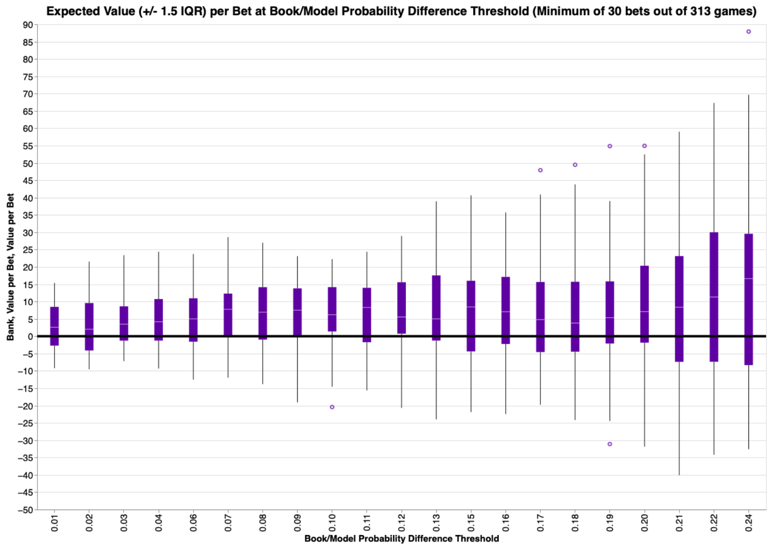

Figure 7: Evaluation on thresholds that triggered a minimum of 30 bets

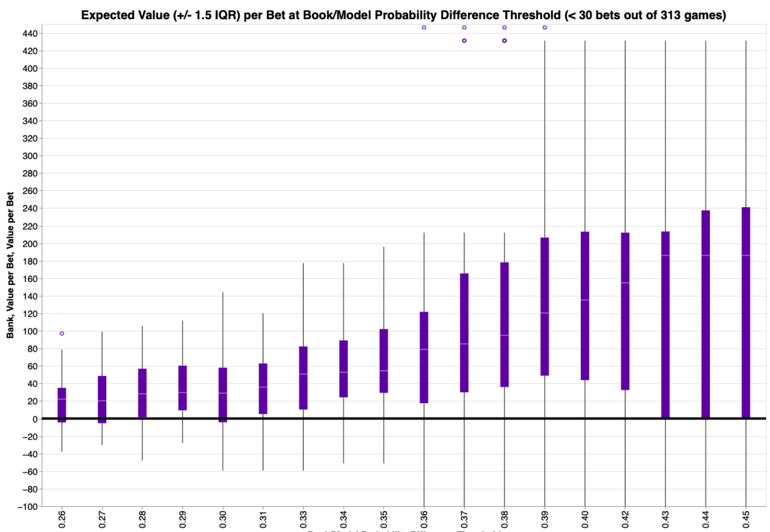

Figure 8: Evaluation on thresholds that barely triggered a bet

Table 2:
Results of 50 Randomly Shuffled Framework Evaluations

Threshold Range	Expected Value(USD) per Bet (+/- 1 SD)	Expected Number of Bets (+/- 1 SD)
0.01-0.1	5.04 ± 1.7	211.5 ± 52.53
0.1-0.2	7.26 ± 0.93	80.8 ± 27.8
0.2-0.3	21 ± 9	20.9 ± 9.6
0.3-0.4	88.8 ± 42.1	4.3 ± 2

4 Conclusion

Overall, this was a challenging project in all parts of the data science pipeline. Whether it was from data collection, to data wrangling/preparation, and finally model training/evaluation I learned many valuable lessons. Neural networks are fascinating function approximators and can be applied to so many different problems. The results were very promising from the model and framework, proving that although the NFL betting landscape is quite efficient, there are still minor inefficiencies that a savvy bettor can take advantage of.

5 Future Work

Going forward, I am eager to evaluate the model in the upcoming season. Additionally, I would like to extend this model to other types of bets and propositions (potentially other sports too).