Decision Making in Betting with Neural Networks

Final Report – Deep Learning (CEE 690.06)

Aaron Williams

Due: 12-11-2018

Abstract:

The primary question that we're going to ask is if we can create a neural network algorithm that once trained can make a statistically feasible profit in sports betting. We're using college football statistics and outcomes. There are numerous bets you can take, but this project will attack the spread, which is an estimated point margin at the end of the game between the two teams which can be bet on either side of. The way will we do this is by estimating the point differential and Vegas spread for a given game. The next step will be to further develop the neural network to tell us two things, which direction we should bet in, and if it the bet is safe. The obvious result of this work would be free money, or perhaps on the flipside, a look at how betting lines can be estimated and perhaps improved.

Overview:

On the year of 1992 a federal ban was placed on sports betting with the main concern being that it was both amoral and would lead to scandals related to fixing games. However, just this year, that ban has been removed and once again in all 50 states the public has the right to place wagers on their favorite teams. My goal is to develop an algorithm that approaches sports betting in a manner that would produce favorable outcomes.

This is a well attacked problem, so I hope to have a unique approach by modeling both the sports matches and the methodology by which betting margins are set. When Vegas (or another group) sets a scoring spread (how much a team is predicted to win by) for a game, they're not always stating explicitly what the expected outcome is, but rather, they're setting a margin which is likely to produce the most profit. By modeling the spread as well as our prediction on the scoring margin using data acquired in relation to our sports teams, we can produce a set of values which make a prediction more accurate.

As early stated, this is a well attacked problem. Vegas tends to be conservative in releasing how they determine betting margins to the public as that would be tipping their hand to potential exploitation. There are basic methods for selecting a margin bet, such as those described in [this article](). Here they describe using different models to beat the spread by identifying over and under estimations in team strength by Vegas. This is a very simple methodology, but the author employs statistics

COLLEGE FOOTBALL WEEK 3 GAMES @WILLIAMHILLUS	CURRENT POINT SPREAD (9/13/18) @WILLIAMHILLUS	% of total # of point spread tickets @WILLIAMHILLUS	% of total point spread $'s wagered @WILLIAMHILLUS
BOSTON COLLEGE vs WAKE FOREST	BOSTON COLLEGE -4.5 (-110)	71% BOSTON COLLEGE	63% BOSTON COLLEGE
GA STATE vs MEMPHIS	MEMPHIS -27 (-110)	84% MEMPHIS	97% MEMPHIS
UTEP vs TENNESSEE	TENNESSEE -30.5 (-110)	94% TENNESSEE	98% TENNESSEE
BALL STATE vs INDIANA	INDIANA -14.5 (-110)	70% BALL STATE	65% INDIANA
TEMPLE vs MARYLAND	MARYLAND -15.5 (-110)	94% MARYLAND	98% MARYLAND
UMASS vs FLORIDA INT	FLORIDA INT -4.5 (-110)	52% UMASS	53% FLORIDA INT
MIAMI FLORIDA vs TOLEDO	MIAMI FLORIDA -10.5 (-110)	84% MIAMI FLORIDA	81% MIAMI FLORIDA
HAWAII vs ARMY	ARMY -6 (-110)	68% HAWAII	73% HAWAII
KENT STATE vs PENN STATE	PENN STATE -35 (-110)	80% PENN STATE	92% PENN STATE
OLD DOMINION vs CHARLOTTE	OLD DOMINION -2 (-110)	73% OLD DOMINION	61% OLD DOMINION
FLORIDA STATE vs SYRACUSE	FLORIDA STATE -3 (-110)	64% SYRACUSE	72% SYRACUSE
OKLAHOMA vs IOWA STATE	OKLAHOMA -18 (-110)	84% OKLAHOMA	91% OKLAHOMA
TROY vs NEBRASKA	NEBRASKA -11.5 (-110)	75% NEBRASKA	92% NEBRASKA
RUTGERS vs KANSAS (NV ONLY)	KANSAS -2.5 (-110)	76% RUTGERS	97% RUTGERS
GEORGIA TECH vs PITTSBURGH	GEORGIA TECH -4.5 (-110)	71% GEORGIA TECH	62% GEORGIA TECH
VANDERBILT vs NOTRE DAME	NOTRE DAME -14 (-110)	52% NOTRE DAME	57% VANDERBILT
OHIO vs VIRGINIA	VIRGINIA -6 (-110)	67% VIRGINIA	71% OHIO
SMU vs MICHIGAN	MICHIGAN -35.5 (-110)	75% MICHIGAN	92% MICHIGAN
BOISE STATE vs OKLAHOMA STATE	OKLAHOMA STATE -2.5 (-110)	62% BOISE STATE	85% BOISE STATE
BYU vs WISCONSIN	WISCONSIN -22 (-110)	74% WISCONSIN	92% WISCONSIN
MIAMI OHIO vs MINNESOTA	MINNESOTA -13.5 (-110)	79% MINNESOTA	56% MIAMI OHIO
LSU vs AUBURN	AUBURN -9.5 (-110)	63% LSU	57% LSU
SOUTH FLORIDA vs ILLINOIS	SOUTH FLORIDA -10 (-110)	92% SOUTH FLORIDA	99% SOUTH FLORIDA
CENT MICH vs NORTHERN ILLINOIS	NORTHERN ILLINOIS -14 (-110)	60% NORTHERN ILLINOIS	64% NORTHERN ILLINOIS
GA SOUTHERN vs CLEMSON	CLEMSON -34 (-110)	61% CLEMSON	64% GA SOUTHERN
NEW MEXICO vs NEW MEXICO ST	NEW MEXICO -4.5 (-110)	68% NEW MEXICO	75% NEW MEXICO STATE
TULANE vs UAB	TULANE -3.5 (-110)	60% UAB	62% TULANE
DUKE vs BAYLOR	BAYLOR -6 (-110)	65% DUKE	53% BAYLOR
COLORADO STATE vs FLORIDA	FLORIDA -20.5 (-110)	81% FLORIDA	98% FLORIDA
HOUSTON vs TEXAS TECH	HOUSTON -1 (-110)	96% HOUSTON	99% HOUSTON
TEXAS SA vs KANSAS STATE	KANSAS STATE -21.5 (-110)	50% TEXAS SA/KANSAS ST	67% TEXAS SA
NORTH TEXAS vs ARKANSAS	ARKANSAS -7 (-110)	59% NORTH TEXAS	78% NORTH TEXAS
SAN JOSE STATE vs OREGON	OREGON -41 (-110)	79% OREGON	82% OREGON
EASTERN MICHIGAN vs BUFFALO	BUFFALO -3.5 (-110)	57% BUFFALO	85% EASTERN MICHIGAN
OREGON STATE vs NEVADA	NEVADA -3.5 (-110)	61% OREGON STATE	94% OREGON STATE
ALABAMA vs MISSISSIPPI	ALABAMA -21 (-110)	82% ALABAMA	90% ALABAMA
ARKANSAS STATE vs TULSA	ARKANSAS STATE -1.5 (-110)	69% TULSA	99% TULSA
TEXAS STATE vs SOUTH ALABAMA	SOUTH ALABAMA -10.5 (-110)	73% TEXAS STATE	99% TEXAS STATE
MD TENN STATE vs GEORGIA	GEORGIA -33 (-110)	98% GEORGIA	99% GEORGIA
MISSOURI vs PURDUE	MISSOURI -6.5 (-110)	80% MISSOURI	91% MISSOURI
AKRON vs NORTHWESTERN	NORTHWESTERN -21.5 (-110)	56% AKRON	81% NORTHWESTERN
UL LAFAYETTE vs MISSISSIPPI STATE	MISSISSIPPI STATE -33 (-110)	94% MISSISSIPPI STATE	99% MISSISSIPPI STATE
UL MONROE vs TEXAS A&M	TEXAS A&M -26.5 (-110)	73% TEXAS A&M	75% TEXAS A&M
W KENTUCKY vs LOUISVILLE	LOUISVILLE -23 (-110)	67% LOUISVILLE	55% LOUISVILLE
USC vs TEXAS	TEXAS -3.5 (-110)	53% TEXAS	64% TEXAS
OHIO STATE vs TCU	OHIO STATE -13.5 (-110)	66% OHIO STATE	53% TCU
WASHINGTON vs UTAH	WASHINGTON -6.5 (-110)	74% WASHINGTON	79% WASHINGTON
FRESNO STATE vs UCLA	FRESNO STATE -2 (-110)	50% FRESNO ST/UCLA	51% FRESNO STATE
ARIZONA STATE vs SAN DIEGO ST	ARIZONA STATE -5 (-110)	83% ARIZONA STATE	76% ARIZONA STATE

Figure 1 - Example of Spreads (also includes percentage of bets cast per side)

A more intricate description of the problem is identified in this paper. Here they propose a method for high volume betting using a NN along with describing how an explicit understanding of the problem is required. They talk about adjusting loss characteristics to help the algorithm model for confidence in order to accurately characterize risk.

One last kernel worth looking at was this one. This is a Kaggle project on betting where they describe how the problem was modeled. Initially, they go into detail on naïve betting strategies before detailing a method of modeling confidence in betting. They use a simple formula for determining the confidence interval by simply using their own confidence over the betting confidence (or the inverse of that). The author asserts that they believe that the line is established based on popularity over prowess in a lot of cases.

Data Description:

We collected the data from several different sources. The goal was to have a lot of relevant statistical information related to college football. Historical rankings for this sport were difficult to come by, so I selected a back-end data approach that I felt would allow the algorithm to do most of the heavy lifting of association and comparison. An easy method if the data was available would be to grab the ELO or overall rankings for each team (like KEMPOM in college basketball), but I could only find the historical rankings of the top 25 teams.

First, let's talk about the statistics (and betting) data we collected from this source. This source provided game statistics data back before the 2001 date that I collected from, but only stored betting data back to 2003 which is where I drew my cutoff. We collected metrics for statistics per game:

Rushing Yards per game, Rush Attempts per game, Passing Yards per Game, Passing Attempts per game, Passing Completions per game, Fumbles per game, Interceptions per game, Final Score

For each of these statistics, we'll be storing the amount earned by the team, the amount earned by their opponent, and the average earned by their opponent over the last 10 games. The final vector for a team for a game will look something like this:

Statistics for particular team = [RY , RY by opp, avg RY by opp, RA, … , avg score by opp]

We'll talk more about what we'll do after that in pre-processing. From the same source, we're also getting our betting information. This includes the spread for each team and the over/under, which we're storing in case we decide to do anything with it later.

I also wanted to nab some recruiting information, which is not as difficult as you might think. I went on to rivals and was able to simply copy the information off the webpage before scripting it out of a .txt. The information I got out of this is as follows:

Number of Recruits, Number of 5-stars, Number of 4-stars, Number of 3-stars, Average Ranking, Overall score (as determined by Rivals)

Finally, to make sure that we included only CBS schools in our recruiting and statistics metrics, I used this and this source. I didn't use those sources for anything besides developing my team map strings, though they also have some other good CFB statistical information.

Sorting:

One important thing to mention is that there were several errors in the data sources I used, particularly with the relation between betting information and game statistics. In some cases, there were duplicated matches or even simply wrong information. A team's participation was determined by a key mapping with several versions of the same team name on each of these interconnected documents. Games for each team were stored separately for both betting information and statistics.

Once both sets were compiled, each team's games were iterated through in order to verify that information was available for a given game in both sets. To verify that two games were the same, the year, opponent, and general whereabouts of the day scalar were compared. If information was included in one set and not the other, the game was ignored and removed. This data was not used as I noticed that in some of the statistical sets a single team was shown to be playing two teams in one week, meaning that the data was wrong. While there are still issues with this method, it eliminates potential non-concurrency issues.

Pre-Processing Methods:

Beside the day scalar which was already [0,1] bound, all statistical and recruiting data was 0-normed using the mean and standard deviations from the training set. This training set includes all data from 2003-2014. The reason we don't include 2015-2017 in determining the norm is because it's the test dataset and we must assume that that information is known.

In order to build our data points, we use 2D arrays which include vectors of past games for each team. The general idea is that this information can establish trends for both teams if viewed in 2-dimensions. Here is an example of a built datapoint:

[[last game statistics for visiting team, last game statistics for home team],

[2nd to last game statistics for visiting team, 2nd to last game statistics for home team],

[…],

[10th to last game statistics for visiting team, 10th to last game statistics for home team]]

This is included as Statistical Data in figure 2. In our recruiting data, we're including several of the past years for each school. Our other info category is completely devoted to a day scalar which was computed by looking the relation of the date of a particular game and how far along in the season it is. Potentially, margins might be adjusted based on when in the season the game was played. 0 would be the first game of the year, and 1 would be the last.

Testing Methods:

For the purpose of testing, the dataset was divided into a training and test set. As previously mentioned, the training set includes yearly data from 2003 to 2014 while the test data set includes data from 2015 to 2017. We'll select things like iteration count and best models by analyzing the loss characteristics of the test data and making an approximation on convergence. There are three steps to testing these methods: Determine parameters, evaluate different models against the spread and actual scoring differential, and do final testing using predictions and confidence determinations.

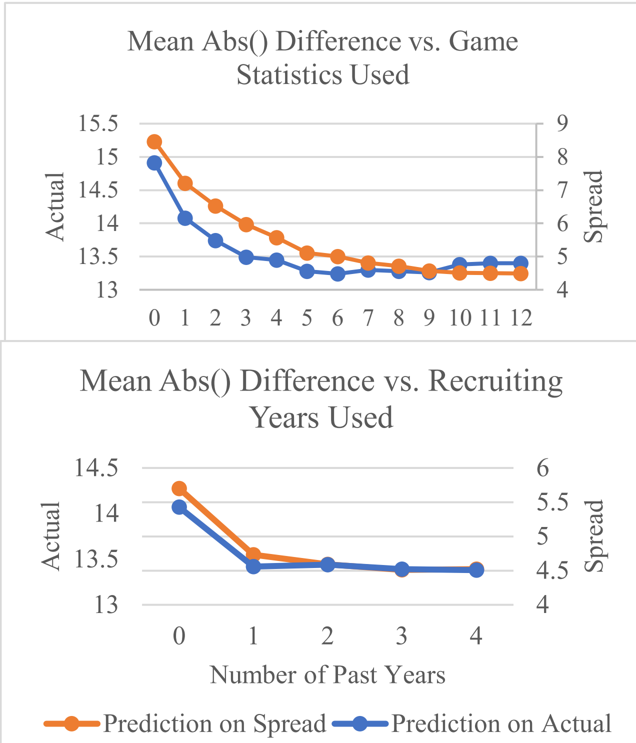

Figure 2 – Past Information inclusion

Determine parameters:

The main parameters which were focused on include the number of past statistical games and recruiting years per datapoint. Not only will this show that each of these sets of data are relevant, but they'll indicate how effective including additional years and games might be. In order to make these determinations, we'll use a basic 2-layer Fully Connected NN setup with raw vectorized data (the same one that we'll use later to see if convolutions are effective). Each layer has no activation function as we're attempting regression.

We're using two separate regression estimates to characterize our data. A prediction of the spread, and a prediction of the actual score. For the number of games, you can see that against the actual score, the evaluations taper off after 6 games, whereas for the spread, correllation improves all the way up until 10 games in the past. For past recruiting years, the data generally levels off at around 3 years, but we'll include 4 as that's how long college players generally stay.

Using Convolutions?

The idea for the convolutions was that we can create a two-dimensional representation of our data by doing row connected layers (as in, converting our 10 50-point vectors into some collection of 2D information). A filter on those rows connected layers would detect patterns in game statistics between games which would lead to a better interpretation of the team's current performance. We use mean pooling in this case as we're not attempting to identify specific features, but perhaps indications of positive or negative trends in the team's game.

We'll test the performance of this model against a simple 2-layer NN, and both models against the Vegas spread that was given. We'll also be looking at importance metrics for each part of the Convolutional NN to identify exactly how effective the convolutional aspects are.

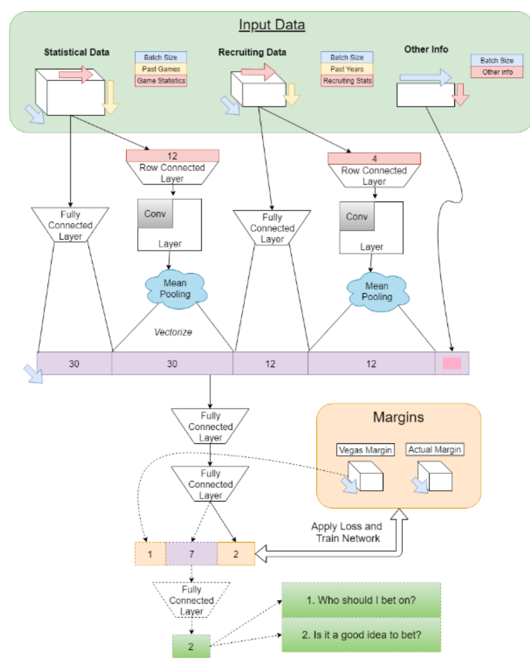

Figure 3 - Convolutional NN

Final Testing, do we make money?

We'll create a final NN approach to the problem. I'd initially anticipated using the Convolutional approach, but this was later changed due to the matched performance of the 2-layer NN. I did keep row convolutions, however, as I believe that they were responsible for the minor improvements in performance with the Convolutional Network over the 2-layer NN. For this testing, we're going to have two training steps. One to optimize predictions given information from the dataset, and the other to develop a confidence metric by analyzing the results of the bet. We can also develop an additional confidence metric by looking at how far the initial prediction is from 0.5, the even point of the sigmoid function.

Finally, we make predictions on the test set. We can use our confidence metrics to rank which games we should bet on. In an ideal situation, even if our betting is ineffective, if our confidence metric is good, we can take sure bets and make money if we select only the games we're most confident in.

Convolution Results:

2-Layer FC NN	Betting Spread	Actual Difference
Prediction on Spread	Avg ± of 4.52	Avg ± of 13.25
Prediction on Actual	Avg ± of 5.24	Avg ± of 13.38

Table 1 – 2-layer NN results

With Convolutions	Betting Spread	Actual Difference
Prediction on Spread	Avg ± of 4.55	Avg ± of 13.24
Prediction on Actual	Avg ± of 4.92	Avg ± of 13.25

Table 2 - Convolutional NN results

In order to evaluate our NNs, we must first establish baselines. The lower bound is 16.83, or the average difference from 0 that is represented in the final margin of scoring for any game. On average, a team that wins does so by 16.83 points. The Vegas spread reduces that prediction to 12.55, which seems insignificant. That value is the average difference between the actual scoring margin and the spread. Essentially, we're only moving 4 points towards the actual score using the Vegas margin.

The results shown in Table 1 and 2 indicate that neither NN performs as well as the guessed spreads, topping out at 13.24 as an estimate for average difference from the actual margin. However, we can accurately guess what the Vegas spread will be to a much smaller margin. We also see that the Convolutional NN only improves results a small bit. One thing that's worth noting is that our characterization of spread and the actual score are very similar, and in some cases our characterization of spread performs better than that on our actual prediction.

Flat stats	Pool stats	Flat recruit	Pool recruit	Day Scalar
±4.67, ±13.23	±4.51, ±13.25	±4.62, ±13.27	±4.57, ±13.26	±4.56, ±13.25

Table 3 – Convolutional NN results

Finally, let's look at the results in Table 3 given that we remove some part of our convolutional vector. None of these show a significant deterioration in actual difference determinations. This means that our Convolutional layers aren't providing much in the way of benefit, but they also aren't hurting the NNs ability to predict the scoring margin.

Earning Results:

Now that we've established our spread predictions as worse than the standard, we move on to attempting to use the information that we do have to earn a profit. As a further breakdown of the problem, in order to make money on spread betting, you need to be correct 52.4% of the time. This is equated from the cost of the bet, $110, over the sum of the potential win and loss, $110 and $100. We are not trying to find a better estimate than the spread but identify where the spread might be wrong using our statistical data. Even with a poor predictor, if we have a good confidence measure, we can find the handful of games that we're sure are being estimated poorly and bet on those.

Our final architecture is our former data with the spread (and over/under) included with row connected layers and then a 2-layer sigmoid NN. Over several models, the average final accuracy was around 51.5%. As this isn't high enough to make money, we need to use our confidence metrics.

Our Dependent metric is derived from initial guesser whereas the independent one is trained on a NN with all our former data, the spread, and the result of the determiner. The independent metric is trained on whether the guess was correct. As shown, both don't perform well, and one might perform better than the other without any consistency. They do mostly trend towards early earnings, but again, not with the consistency you'd put money on.

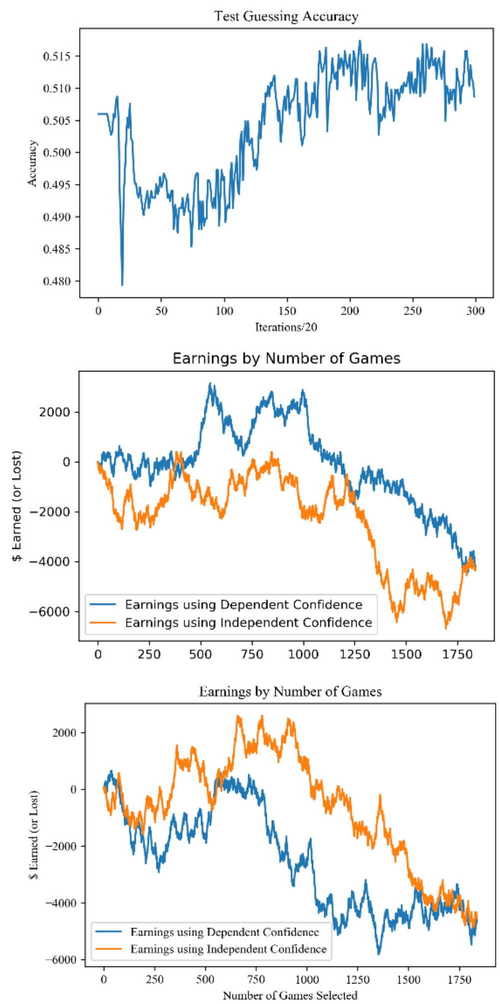

Figure 4 - Results Profit Estimation

Conclusions:

There are several sources of data that were missing for this project. Some of the main ones like coaching and injuries can have a profound effect on the scoring result of an upcoming game. Even knowing something like, this team is starting a rookie QB this season, is very important. While it's easy to find information for one set of teams and thereby estimate the results of a single match, it's difficult to do so in retrospect and for every team in FBS. We want consistency in a NN algorithm. I think the inclusion of some sort of injury information, coaching information, and perhaps other metric guesses like Elo would be enough to make more accurate estimates and turn a consistent profit. The main problem comes down to data insufficiency, and it was difficult enough to compile recruiting and statistical data, especially knowing that several holes existed in that dataset.

Future Work:

Moving forwards, I would initially see if results were better at estimating the over/under. It's possible that these same NN methods could still form a stable betting strategy. I also believe that an RNN application could be useful for storing games result by result. An RNN application would be difficult though with multiple teams as you'd need to account for information from both. Also, adjusting results retrospectively (say a team that was estimated to be good is bad) would be difficult, not that this current algorithm does so. I was a little disappointed with the results of this project. I was hoping that convolutions or the confidence estimates would turn out better.

Honor Code:

I adhered to the honor code in the completion of the assignment.

References:

Chernoff, A. (2017, August 30). How To Create And Use NFL Power Ratings To Beat The Point Spread. Retrieved from https://medium.com/@adamchernoff/how-to-create-and-use-nfl-power-ratings-to-beat-the-point-spread-3fa4c3ecdc22

A machine learning framework for sport result prediction. (2017, September 19). Retrieved from https://www.sciencedirect.com/science/article/pii/S2210832717301485

Thomas, Eduorard. "Beat the Bookmakers with Machine Learning (Tennis)." RSNA Pneumonia Detection Challenge | Kaggle, www.kaggle.com/edouardthomas/beat-the-bookmakers-with-machine-learning-tennis.

http://sports.snoozle.net/search/fbs/index.jsp

https://n.rivals.com/team_rankings/2019/all-teams/football

https://www.masseyratings.com/

https://github.com/mattp88/ncaa/blob/master/teams.csv