How to Beat Vegas Using Tree-Based Models

Jordan Butler

STAT 7940: Applied Analysis Project

Dr. Austin Brown

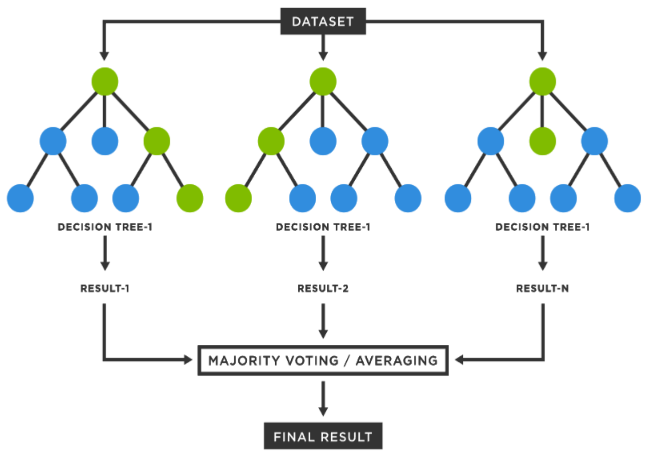
Sports betting is just one of the infinite number of controversial topics that exist in the world today. Many people are in favor of no restrictions on sports betting, while others support a total ban on it. Regardless of where one stands on the issue, something that is widely accepted is that in Vegas, the house always wins. In sports betting, only 3-4% of people profit long-term, largely due to people placing low percentage bets in hopes of effectively winning the lottery. On the other end of the spectrum is standard vig straight bets which are singular bets that have an actual winning probability of 50%. In the NFL (most commonly bet on sports league in the world), two of the most common ones are over/under (predicting total points in game will be more/less than given number) and spread (predicting final score difference between two teams will be more/less than given amount). Because ties are considered pushes (bettor gets their money back), these bets have a 50% chance of being correct. However, the way Vegas gets their edge is by offering both sides of these bets at a 52.38% chance of winning or -110 odds (giving Vegas a 4.76% total house edge). This means that in order to profit from these straight bets, you need to win more than 52.38% of your bets assuming you bet the same amount every time. That extra 2.38% may not seem like much, but it is enough to render 99% of human-based prediction algorithms/systems un-profitable in the long run. By removing the "human" aspect altogether and leaving the predictions and betting styles to statistical modeling software, it IS possible to consistently make money betting on sports and it is all thanks to the power of tree-based models.

Tree-based models are a type of supervised learning algorithm that split data at various points (into branches) where each branch eventually leads to an aggregated prediction based on average or counts. While there are many kinds of tree-based models, there are two main approaches used which are bagging and boosting. Random Forest was used as an example of a bagging algorithm where multiple trees are built independently of each other at the same time with the final row of branches being aggregated at the end for the overall prediction. XGBoost was used as an example of a boosting algorithm where the trees are built sequentially with each correcting the errors of the previous one. With Random Forest being prone to under-fitting and XGBoost being prone to over-fitting, a third prediction setup was constructed by combining the results of the two models. The models were built in R Programming using the nflfastR dataset which contains 300+ columns of data for every play that has been run in the NFL since 1999. The testing data used to teach the model included a simplified 40ish attributes for each game from 2003-2022. These attributes included variables describing the game environment (playing surface, weather, regular season vs playoffs, etc.), the teams themselves (offensive and defensive rankings 1-32 assigned to each team based on EPA and WPA), and the closing Vegas lines for both the spread and the over/under. The completed model taught by said variables was then used to predict the delta (measured in points) between Vegas predictions and actual results for both the over/under and spread lines for every game in the 2023 NFL season (excluding Week 1 due to inability to have offensive and defensive rankings based on previous games).

In addition to the model being used to predict how far off Vegas will be from the actual number, there were a couple of other features added to maximize net profit %. The betting strategy implemented is very similar to counting cards in blackjack. While counting cards, the goal is to bet more money if there are more 10s/face cards left in the deck. Similarly, our goal here is to implement a weighted betting system where the bets are larger if the model is more confident that Vegas is wrong. While there are a myriad of potential betting systems to implement, I kept it simple and decided that for every point the model predicts Vegas is off by, an additional unit would be added to the overall bet. For example, if my unit is $10 and model thinks that Vegas is off on the over/under number for a particular game by 4 points, I would now be betting $50 (4 additional units) instead of the standard 10. Maximizing the profit margins based on confidence is a necessity in any betting system and is infinitely better than flat betting, as will be shown in the betting results for each of the models. While not quite as necessary as weighted betting, an additional aspect I implemented in my model is determining the ideal point confidence threshold at which you should start betting. In blackjack, many players will often sit out and wait until the true count (confidence unit for counting cards) reaches a certain number. Similar to that principle, I tested the win % and net profit % if you only bet on games that the model was x points confident Vegas was wrong by. The ideal betting thresholds and their results will, along with the results of weighted betting vs flat betting, will be detailed first for the Random Forest model then followed shortly thereafter for the XGBoost model.

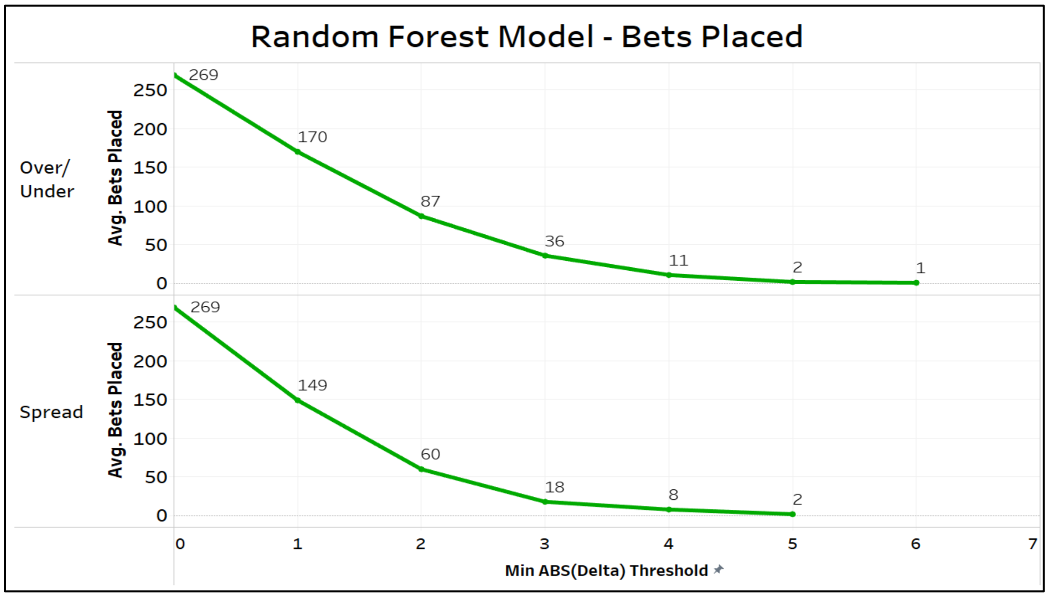
With Random Forest model trees being built independently of one another, they often fail to recognize certain correlations between attributes and can therefore under-fit the model to the data.

Figure 1



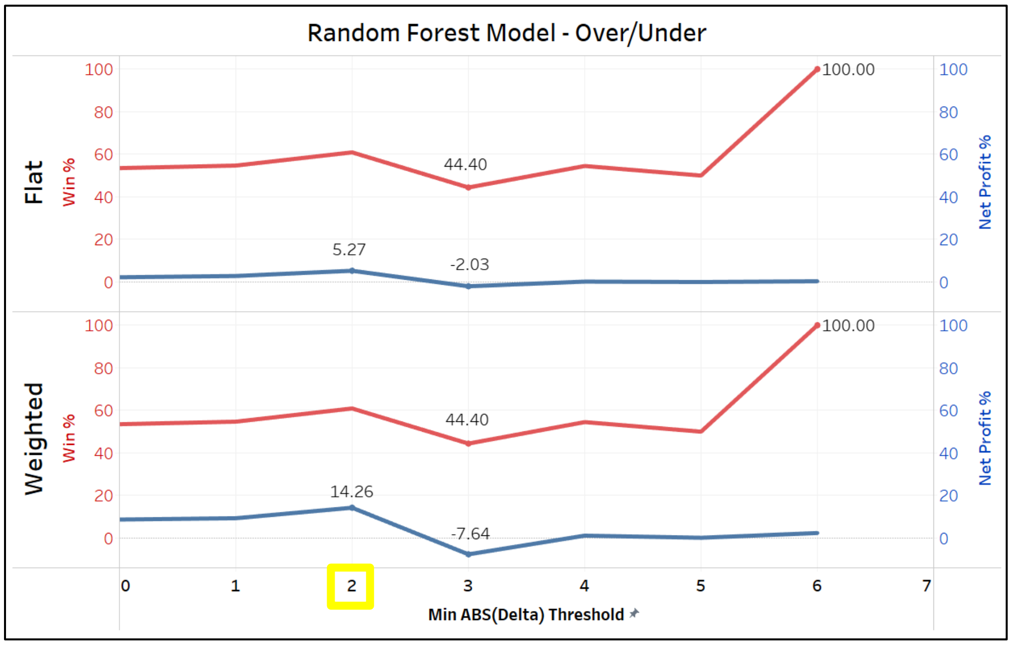
With respect to the sports betting predictions, this led to the model being less confident regarding predictions both with the spread and the over/under. In other words, the predictions tended to be, while still correct more often than not, closer to zero than the actual result. With each point away from zero representing a unit of confidence in a way, the amount of games the Random Forest model predicted Vegas being x points away from the actual can be viewed below in Figure 2.

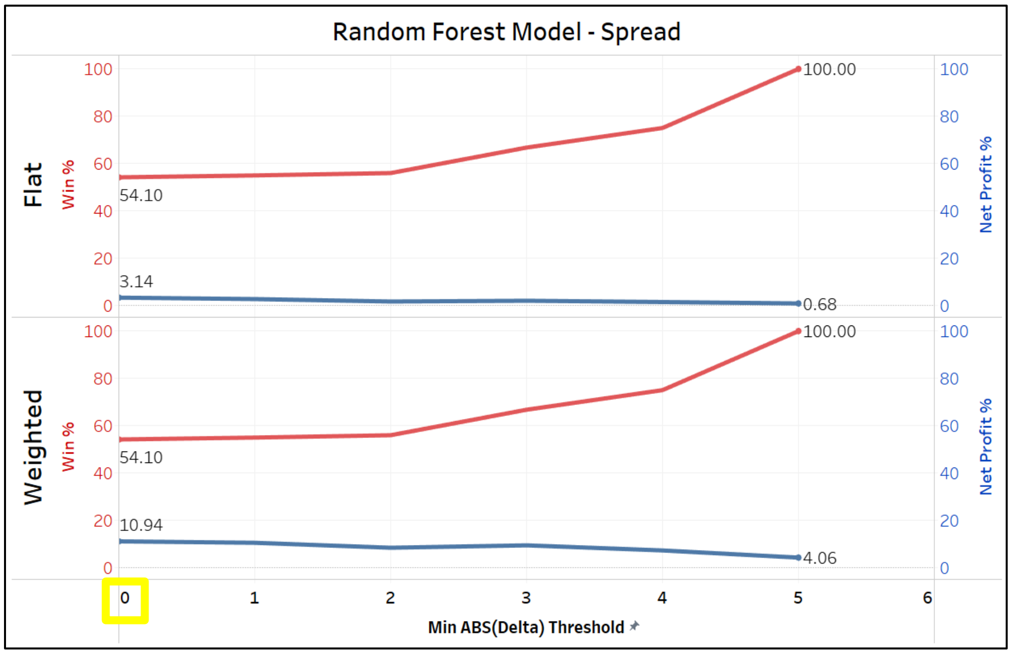
Figure 2



As shown above, there is an expected exponential decrease in the amount of games that the model says Vegas is x number of points away from the actual value, with the maximum value being a singular game where the model thinks Vegas is off by at least 6 points and only on the over/under. It should also be noted that the model is much more confident in predicting the over/under vs the spread which can be seen by comparing the number of bets placed at each level of x between the two bet types (ex. at ABS(Delta) Threshold = 3 the model has twice as many predictions for the over/under as the spread [36 vs 18]). The question is then, at which threshold of betting only games that the model thinks Vegas is wrong by that number of points for either bet type is net profit % at its highest. Those can be shown below in Figure 3.

Figure 3

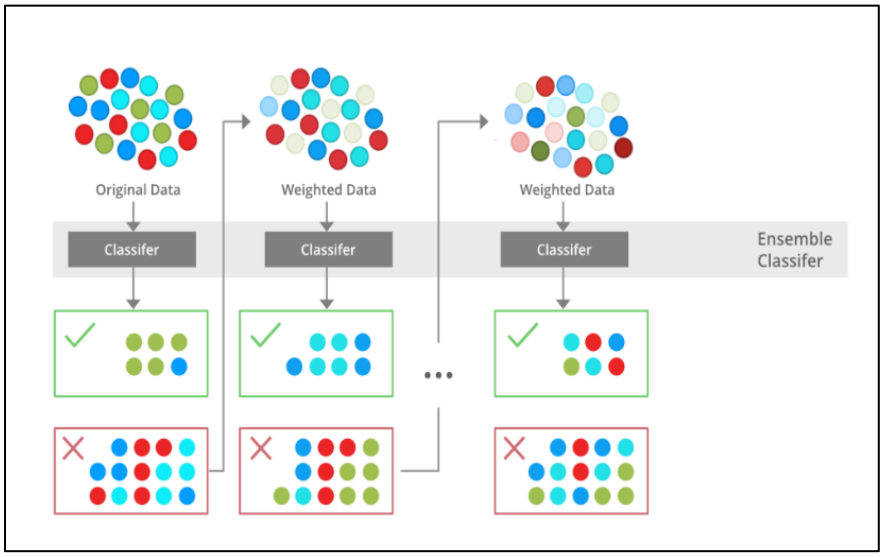




The above figures reveal a couple of key findings about the accuracy of the models. The first things shown is that, contrary to popular belief, is that a higher bet win% does not necessarily signify a higher net profit %. For example, while the model both for over/under and spread has a win % of 100% at a threshold of 5 and 6 respectively, you are only placing 3 total bets therefore the actual amount of money you are winning is relatively restricted. Furthermore, the necessity for weighted betting is revealed when comparing the net profit % between flat vs weighted betting for the two bet types under the Random Forest model. For over/under, the minimum point difference threshold that results in the highest net profit % is 2 for both betting styles. This means that, in order to maximize net profit % while betting over/unders using the Random Forest model, you should only bet on games that the model think Vegas has wrong by AT LEAST 2 points in either direction. However, the difference in net profit % between betting styles is ~ 9%, meaning you will profit 9% more by using a weighted betting system vs flat betting. For the spread predictions, the minimum points threshold that results in maximized profit is actually at 0 meaning that you should be betting the spread on all 269 games between Week 2 and the Super Bowl. However, once again there is a big difference at 7% between the net profit for weighted vs flat betting, further reinforcing its importance. A 14.26% and 10.94% net profit for betting the over/under and spread is a very solid margin that cannot be ignored considering how conservative the Random Forest model is. But how exactly would these results compare with those of the ultra-aggressive XGBoost model?

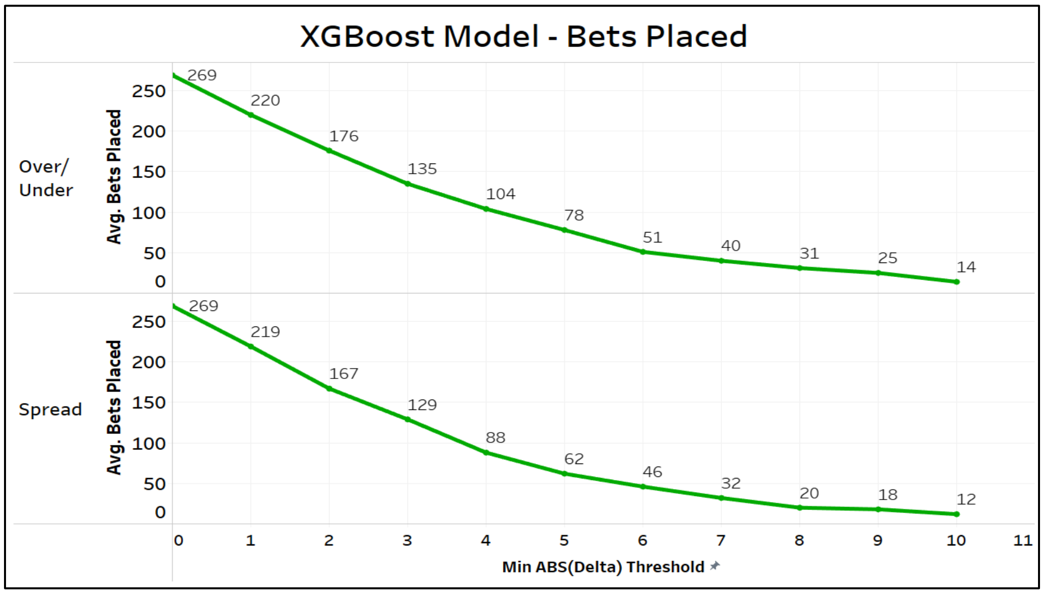
The main difference between the Random Forest model and the XGBoost model is that using XGBoost, the trees are built sequentially with each learning from the inaccuracies of the previous tree.

Figure 4



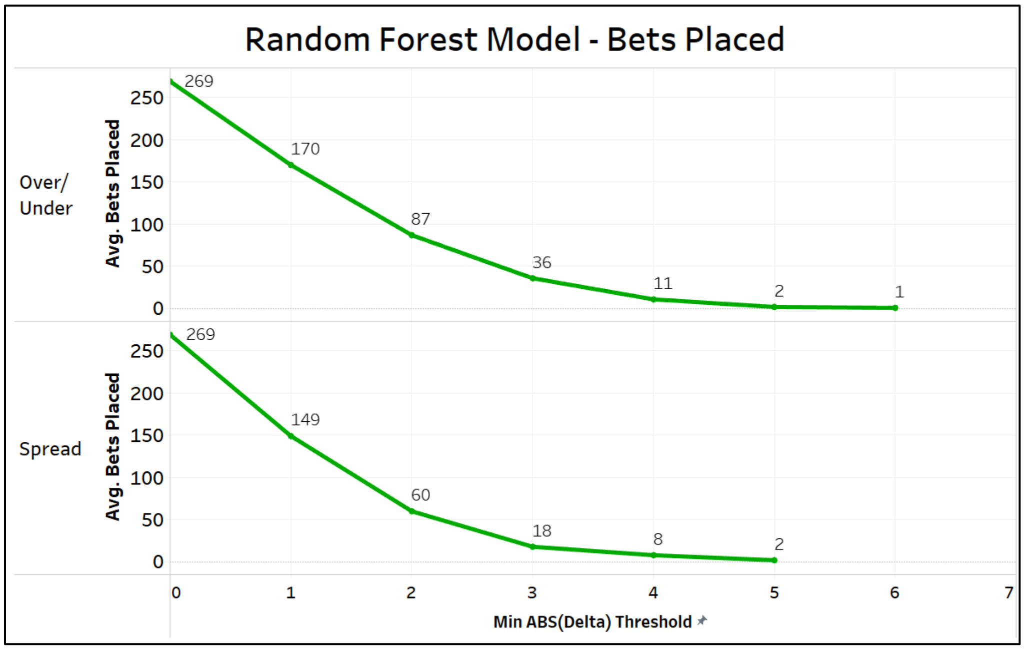
As shown above in Figure 4, following the completion of 1 tree, the weighting of certain attributes are altered. This allows for the model to test for certain factor levels that may consistently result in more convincing results one way or the other. While this does often negate the potential for overlooking the importance of certain variables like what could happen in Random Forest models, this can lead to the XGBoost model over-stressing important on some variables. This isn't necessarily a bad thing in this particular case as it could be over-stressing a super-important variable. So while the model's prediction of whether it think Vegas is higher or lower than the true number may be "correct", it may often times over estimate the amount by which Vegas is off. Due to this matter, the number of games the model has at each confidence threshold is going to be much higher than with the Random Forest model. This effect can be seen below in Figure 5.

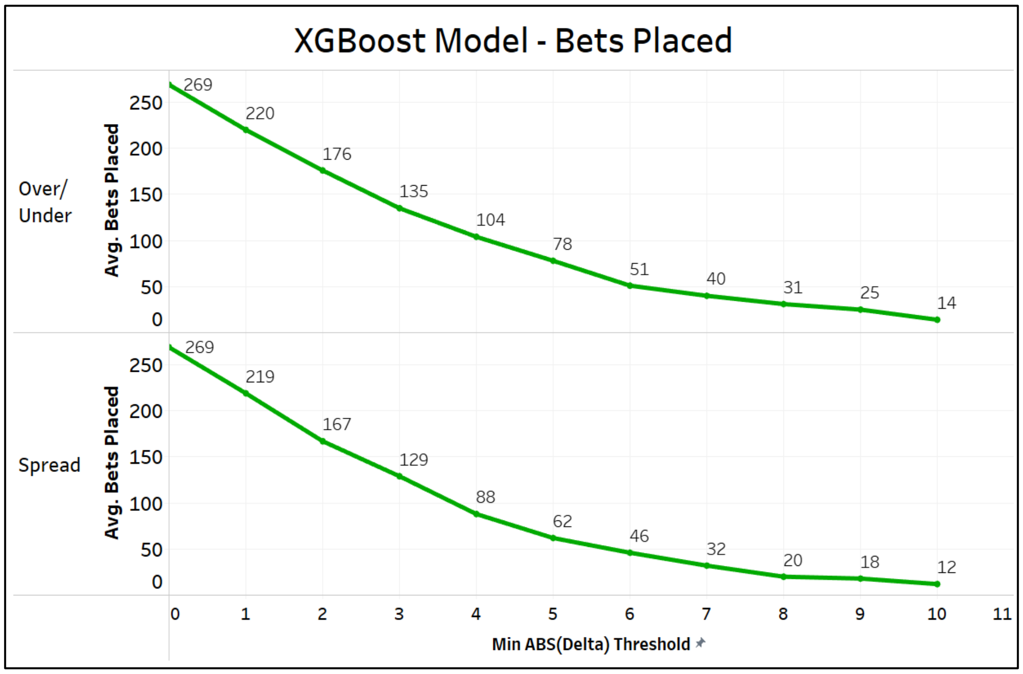
Figure 5



The first thing to observe is the maximum point difference for both of the models. As shown above in Figure 5, the XGBoost model predicts differences all the way up to 10 points with a few even beyond that (aren't included in x axis of graph). This is in comparison to a max of 6 for the Random Forest mode. Furthermore, looking at the difference in number of games at each point threshold, we can see an even larger disparity between the two models.

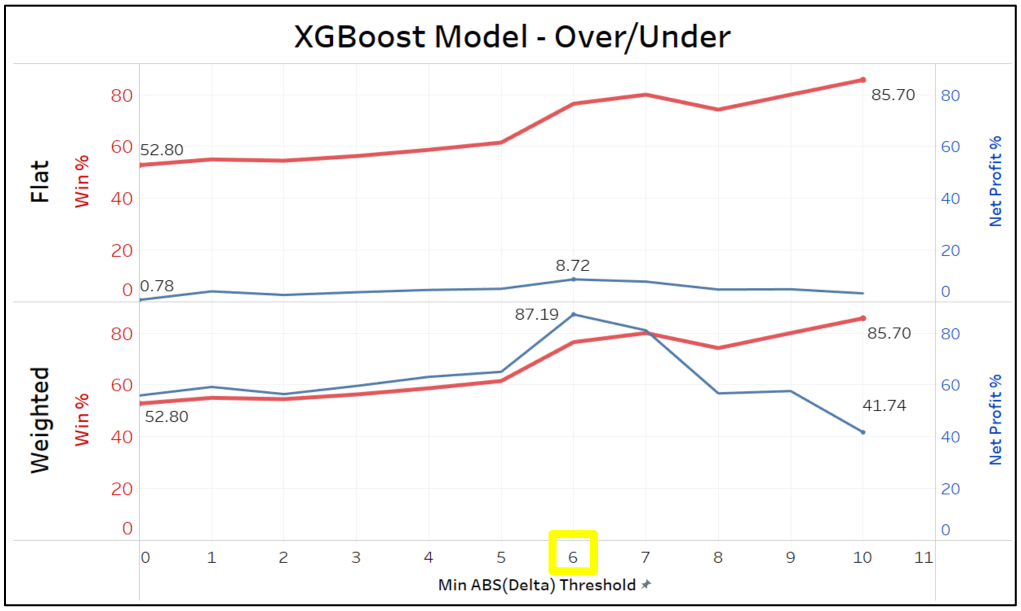
Figure 6

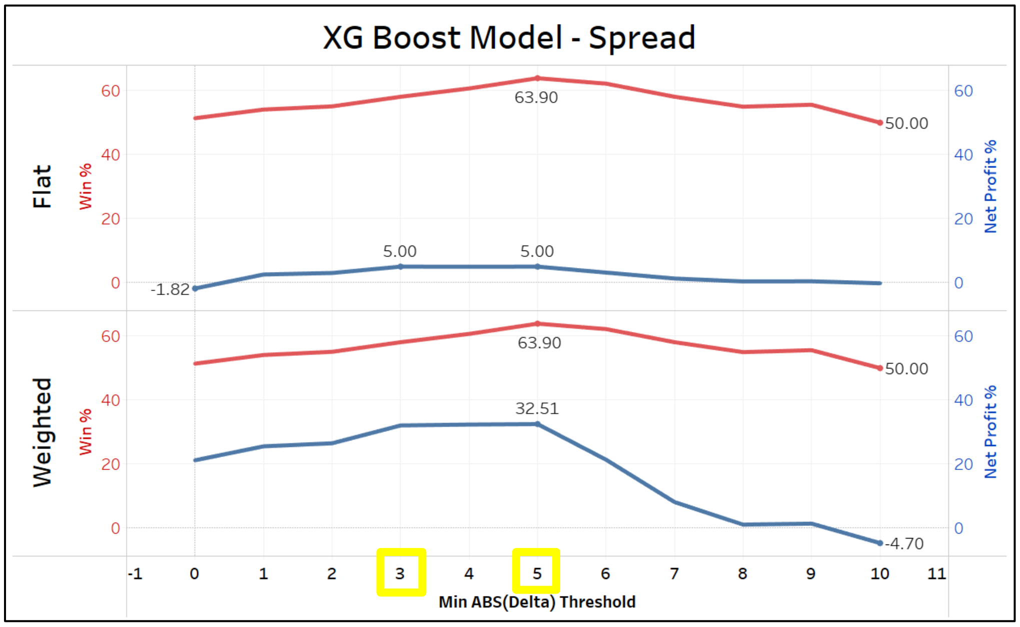




Even at just the 1 point threshold, there is a difference of 50 games between the two models for over/under and 70 games for the spread. That disparity only grows larger as the point threshold increases. Additionally, similar to the Random Forest model, the XGBoost model consistently has a higher mean confidence in betting the over/under vs the spread, solidifying the fact that (as expected) there is much less unpredictable variance in the over/under and is therefore easier to predict than the spread. The accuracy and net profit % of the XGBoost model can be seen below in Figure 7.

Figure 7

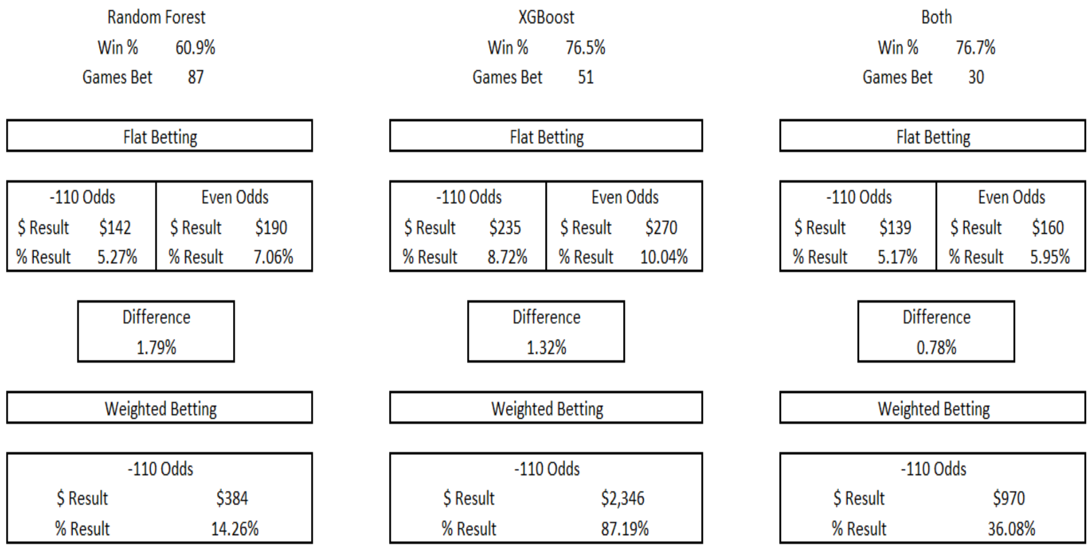




Just as with the Random Forest model, a higher bet win % does not necessarily translate to a higher net profit %. Continuing with the theme of betting games at the point threshold for which the model has the highest net profit %, over/unders should only be bet on games which the model is at least 6 points confident one way or the other, and spreads should only be bet on games which the model is at least 3 points confident in either direction. While the XGBoost model does produce a slightly higher net profit % (less than 1%) at a 5 point threshold vs a 3, the reason for which a threshold of 3 was implemented is because of the combination model. The extremeness of the XGBoost model certainly appears to pay off, with a max net profit % of 32.51% on spreads and a whopping 87.19% on over/unders. However, because at the end of the day this is still gambling and the element of luck has to be considered, one may wonder whether or not the risk of placing significantly more bets is worth the maximized assumed net profit. Because at the end of the day, this model only can maximize the chance of winning it can't guarantee it. Therefore, risk of ruin (risk of losing everything) must be considered although not necessarily able to be measured. However, a way to mitigate this is to take both model into consideration.

Although it may seem counterintuitive, there are certain games in which each of the 2 models predict a different outcome. While this is just the reality of model building, one thing that can be done to help mitigate the differences in predictive capabilities of each model is to look at how well the models may work together. In this situation, that would mean looking at the potential of only betting on games where both models give the same outcome. This can be further tuned by setting thresholds for each model with each bet type to maximize net profit %. This would be the thresholds surrounded by a yellow box in Figure 3 and Figure 7. The results of this combined model for over/under can be shown below in Figure 8.

Figure 8



Random Forest

Win %	60.9%
Games Bet	87

Flat Betting

-110 Odds		Even Odds	
$ Result	$142	$ Result	$190
% Result	5.27%	% Result	7.06%

Difference

1.79%

Weighted Betting

-110 Odds	
$ Result	$384
% Result	14.26%

XGBoost

Win %	76.5%
Games Bet	51

Flat Betting

-110 Odds		Even Odds	
$ Result	$235	$ Result	$270
% Result	8.72%	% Result	10.04%

Difference

1.32%

Weighted Betting

-110 Odds	
$ Result	$2,346
% Result	87.19%

Both

Win %	76.7%
Games Bet	30

Flat Betting

-110 Odds		Even Odds	
$ Result	$139	$ Result	$160
% Result	5.17%	% Result	5.95%

Difference

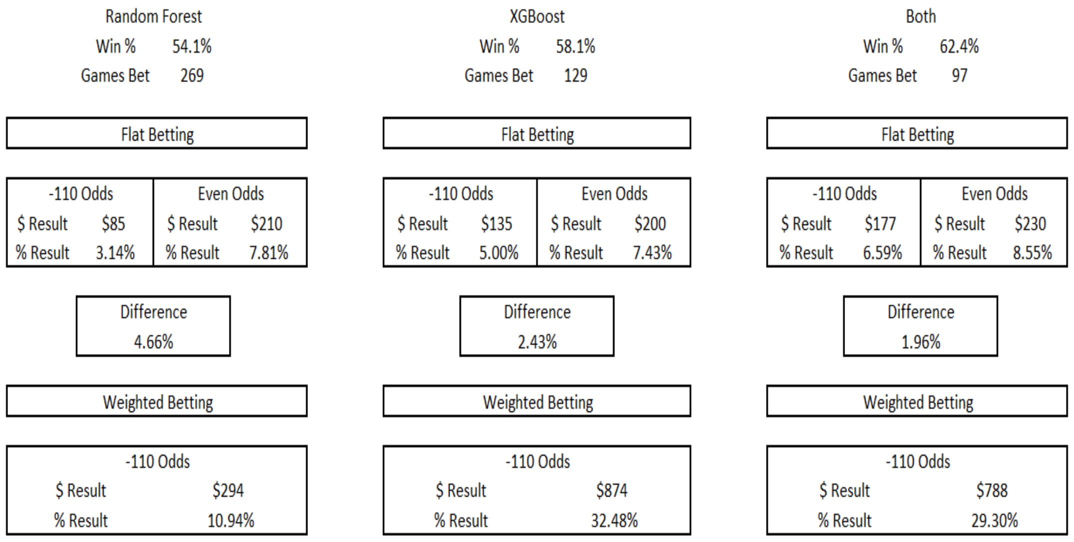
0.78%

Weighted Betting

-110 Odds	
$ Result	$970
% Result	36.08%

As expected, the combining the models results in betting in a significantly less amount of games at only 30/269. However, the bet win % jumps to 76.7% with a net profit of 26.08%. While this is a significantly lower number than the 87.19% using just the XGBoost model, something that can't necessarily be quantified but is certainly present is the fact that much less money is being bet therefore minimizing the risk of ruin. The results are similar when looking at how combining the models performs on betting the spread as shown below in Figure 9.

Figure 9



Random Forest

Win %	54.1%
Games Bet	269

Flat Betting

-110 Odds		Even Odds	
$ Result	$85	$ Result	$210
% Result	3.14%	% Result	7.81%

Difference

4.66%

Weighted Betting

-110 Odds	
$ Result	$294
% Result	10.94%

XGBoost

Win %	58.1%
Games Bet	129

Flat Betting

-110 Odds		Even Odds	
$ Result	$135	$ Result	$200
% Result	5.00%	% Result	7.43%

Difference

2.43%

Weighted Betting

-110 Odds	
$ Result	$874
% Result	32.48%

Both

Win %	62.4%
Games Bet	97

Flat Betting

-110 Odds		Even Odds	
$ Result	$177	$ Result	$230
% Result	6.59%	% Result	8.55%

Difference

1.96%

Weighted Betting

-110 Odds	
$ Result	$788
% Result	29.30%

Similar to combining the models for the over/under, the combined model for the spread resulting in less bets placed with a high bet win %. While the net profit % again isn’t as high as with the XGBoost model, the overall risk of ruin is still lower and much less money is being bet.

In conclusion, both the Random Forest and XGBoost models have shown that it is in fact possible to profit betting on over/unders and spreads in NFL games. The Random Forest model is more of a conservative one while the XGBoost has clearly shown its ultra-aggressiveness. Using both models to bet on games doesn’t necessarily yield as high of a net profit % as the XGBoost, however it is much safer and has a significantly lower risk of ruin. The 2 additional elements to the betting model other than the predictions are the weighted betting and betting thresholds. By implementing weighted bets to the overall betting system and betting more when the model(s) is more confident, the overall net profit % increases significantly. Additionally, not betting until a certain point threshold is met at which the model performs the best drastically improved the net profit % along with reducing the risk taken via reducing the number of overall bets. Overall, each of the 3 models can be used effectively to provide a positive net profit % depending on the goal of the user whether it be maximizing profit, minimizing risk, or somewhere in the middle.