Machine Learning for Sports Betting

Hank Rugg
Department of Mathematics
Theodore Wendt, PhD
June 5, 2024

Abstract

With sports betting becoming more widely legal, the use of machine learning algorithms for improving an individual's odds of placing successful sports bets has increased. In general, applying machine learning algorithms comes with challenges such as data selection, feature engineering, and dealing with time series data. In the context of gambling, it also comes with ethical considerations such as the use of such models to gamble, the accuracy of the model, and the transparency of the model. This research focuses specifically on predicting the total combined score of NBA games. This is directly applicable to the Over/Under bet – "Over" if you believe the combined total score will be above the number set by the sportsbook and "Under" if you believe the combined score will be less than the number set by the sports book. The goal of this research is to create a machine learning model that can accurately predict the total combined score of NBA games.

Introduction

The objective of this project is to create a model that can be used to predict the outcome of the over/under bet. There are three potential outcomes for this bet: over, under, and push. "Over" occurs when the total points scored at the end of the game exceed the line set by the sportsbook. "Under" happens when the total points scored at the end of the game are below the line set by the sportsbook. Finally, "push" arises when the total points scored at the end of the game exactly match the line set by the sportsbook. Push is a relatively rare outcome, as most sportsbooks set the line at half numbers, making exact matches less common. To be profitable, the bettor must be over 53% accurate in their bets due to the payouts by sportsbooks (Jensen, 2022).

Background

There are many types of sports bets, some common types include Moneyline, Spread, and Parlays. Moneyline is based on the outright winner of the game, regardless of the amount of points scored. This is one of the most popular bets and is quite easy to understand. Another bet is the Spread. This is also betting on who will win the game, yet by how much, which is where the spread comes in. Say the Celtics are -11.5 point favorites over the Pistons. This means the Celtics are favored to win by over 11 points. If someone bets on the Pistons to “cover the spread,” this means the Pistons can lose by up to 11 points and that person will still win the bet. Any bets can be combined into a Parlay which is multiple bets combined into one. This means all “legs” of the Parlay must hit for the bettor to win. If at least one of the legs misses, the bet is lost and the bettor does not win any money.

Sports betting has been a contentious topic since its inception, often associated with wrongful outcomes of games due to bribery. The passage of laws, notably the Professional and Amateur Sports Protection Act (PAPSA) in 1992, outlawed sports betting in states that did not already permit it, leaving Nevada as the sole legal state (Bonesteel, 2022). After decades of controversy and illegal betting activities, the Supreme Court deemed PAPSA unconstitutional, allowing individual states to enact their own sports betting laws (Bonesteel, 2022).

In 2018, Delaware was the first state aside from Nevada to introduce sports betting and is home to the first non-Nevadan sportsbook, DraftKings (Bonesteel, 2022). Some form of sports betting is available in over half of the states, with many restricting different forms of betting in efforts to promote safe betting practices (CBS Sports Staff, Bengel, McCariston, 2023). In Montana, sports betting is available through in-person betting at approved locations through the Montana Lottery.

Relevant Work

In the paper, *The Bank is Open: AI in Sports Betting*, Bucquet and Sarukkai use a Long Short Term Memory (LSTM) neural network to predict the outcome of Over/Under bets in the NBA

with a 51.5% accuracy(Bucquet & Sarukkai, 2018). They used data from the 07-08 season to the 18-19 season and included stats such as Points Scored, Assists, Rebounds, and Steals (Bucquet & Sarukkai, 2018).

In the article, *Beating the Books: Using Machine Learning to Make Money Sports Betting,* Jensen uses a Gradient Boosting Regressor to predict the outcome of Over/Under bets in the NBA with a 58% accuracy (Jensen, 2022). Jensen manually collected game data from the NBA website (Jensen, 2022). Jensen also points out that a bettor must be at least 53% accurate in their bets to be profitable (Jensen, 2022). This can vary based on the odds offered but is based on the general -110 odds for Over/Under betting.

Machine Learning Models

Machine learning can be an ambiguous and daunting term. Géron defines it as, "The science (and art) of programming computers so they can learn from data" (Géron, A. 2019). The first widespread machine learning model was implemented in the 1990s– the spam filter (Géron, A. 2019). This model is now quite accurate and classifies an email as "spam" or "not spam". The learning part comes in through the way the model can adapt to different types of spam based on the data it is being fed (Géron, A. 2019). Some problems can use simple methods such as logistic regression to accomplish their goals whereas other problems need more advanced algorithms such as neural networks.

Loosely based on the resemblance of a human brain, the neural network is one of the most advanced machine learning algorithms (Géron, A. 2019). Just as the human brain has connections through neurons, the artificial neural network attempts to imitate this by creating different connections between variables based on different weights (Géron, A. 2019). The components of a neural network include nodes, weights, layers, activation functions, bias, and loss functions. Layers are composed of nodes that contain the weight and activation function. A neural network could look like an input layer of nodes followed by a hidden layer of nodes (referring to the fact that it is neither the input nor output) and finally an output layer. An example of a basic neural network is shown below.

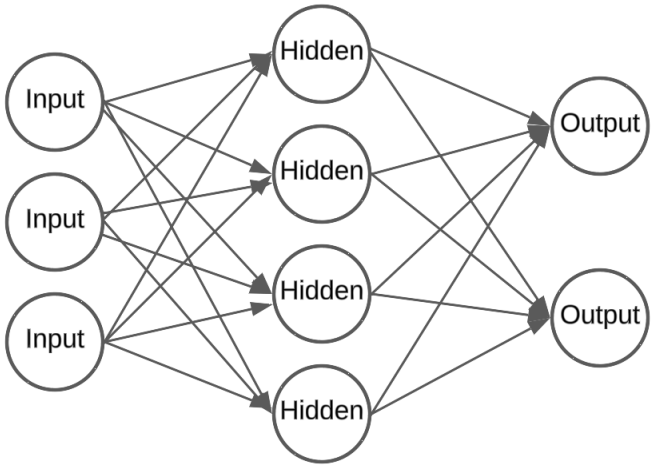

Diagram 1: Simplified neural network

The equation used for each node is shown below.

$$y = (xw + b)$$

Equation 1: Computation for each node

Where x is the output from the previous node, w is the weight associated with that node, and b is the bias. The weights are initialized at random values and are honed through the process of backpropagation as the model trains.

The Data

There are several challenges associated with predicting the outcome of the Over/Under bet. One of these challenges is finding accurate data. Any machine learning model is only as good as the data fed into it. If accurate and relevant data cannot be found, making accurate predictions will be difficult. For this specific problem, game data from historical NBA games along with historical sports odds is used. Another challenge is combining data from multiple sources. One set of data from NBA games was used to supply game stats and another set from historical sports odds to supply Over/Under lines.

The data used in this model specifically comes from regular season NBA games from the 2021-2022 and 2022-2023 seasons. The stats from each team's two most recent games are utilized to predict the outcome of the current game. This includes metrics such as:

- Team ID
- Date

- Matchup
- If they won or lost the game
- Amount of wins
- Amount of losses
- Win Percentage
- The number of minutes the game lasted
- Field goals made
- Field goals attempted
- Field goal percentage
- Three pointers made
- Three pointers attempted
- Three point percentage
- Free throws made
- Free throws attempted
- Free throw percentage
- Offensive rebounds
- Defensive rebounds
- Total rebounds
- Assists
- Steals
- Blocks
- Turnovers
- Personal fouls
- Points scored

This data is provided by a free API that supplies statistics from historical NBA games titled nba_api available on Github written by Swar Patel (Patel, 2024).

Additionally, historical odds data is used which provides the Over/Under line for each NBA game from the 2021-2022 and 2022-2023 seasons. This data includes:

- Date
- Matchup
- Over/Under line

Odds data is provided by a proprietary API that supplies historical odds data for sports bets titled odds-api (Odds API). The two sources of data were combined using the Date and the Matchup.

Implementation Technology

The data is modeled in Python using different machine learning specific libraries such as

- Scikit Learn

- Pandas
- Numpy
- Keras
- Matplotlib

Data Preprocessing

This data is used to create multiple models and compare the accuracy with each one. One of these models is a neural network, and therefore we should modify the data before all models are tested to be fair to all models. Neural networks are highly sensitive to the input data and therefore require extra attention to ensure it is processed correctly. To prepare our data for modeling we will encode our data. This is a process of turning categorical variables into boolean representations so the models can use this data. This is helpful for categories such as Team ID and will allow us to use the team as a predictor. This is done using Pandas' "get_dummies" function. The Date and Matchup are removed since this could introduce some snooping of the data which could give hints to our model that is not desirable. For example, if the model knows that, hypothetically, on February 24 the Celtics only scored 74 points, the model could infer that something, such as an injury to a key player, happened in their previous game on February 22 and predict a lower score for that game. Although this is a hypothetical and very specific situation, if our model is going to be as realistic as possible, the possibility of data snooping needs to be minimized. Instead of using the explicit date as a predictor, the two most recent games from each team are used to capture how the team is performing at that point in the season. The encoded data for the Team ID is used along with the data from each team's two most recent games which provides 157 predictors.

Next, the data is normalized. The Standard Scalar is used and is available through SciKit Learn. This centers each variable around 0 and creates a uniform variance for each variable (Keras Team). Normalization is a crucial step as it ensures that all variables are on the same scale, preventing any single variable from dominating the training process (Keras Team). This also helps prevent our gradient descent algorithm from getting stuck in local minima by ensuring uniformity across all variables.

Finally, the data is split into testing and training sets. This is a standard practice in machine learning development, allowing evaluation of the model's performance on unseen data. 20% of the data is randomly selected and will serve as the testing data, while the remaining 80% will be used to train the model. The test-train-split is used and is provided by Scikit Learn (Keras Team). This training set is what the model uses to establish the parameters and patterns of the model. Once trained, the model's performance is assessed on the data it did not use for its training and has never seen before– the testing data. This will provide an accurate estimation of its effectiveness and whether it is an accurate model.

Model Selection

When determining which algorithm to use for the machine learning predictions, it's common practice to experiment with various models to identify the most effective one. The choice of model depends on its accuracy compared to other models, as well as the ease of interpretation. This is a classification problem therefore all the algorithms need the ability to perform classification.

Null Model

Initially, a null model is developed which serves as a baseline for evaluating the effectiveness of subsequent models. For this problem, the mode of the training data is employed. Using the mode as the null model is standard for classification problems, as it represents the simplest possible approach. “Over” is predicted for all observations in the testing set. Betting over for all observations in the testing data yields a 50.66% accuracy. This accuracy rate will serve as the baseline for assessing the performance of the machine learning models.

Decision Tree

The decision tree is a model that makes classifications based on the values of its variables. Essentially, it creates a flow-chart that it will follow to classify each observation. The decisions it makes are based on whether the value of a variable is greater than or less than a randomly set criteria. The model will make a decision at each node it encounters. There are a set of hyperparameters that can be tuned to make better predictions. These include the maximum depth of each tree, the amount of samples needed in a final node, and the amount of samples needed in a non-final node (Géron, A. 2019). To get the optimal set of hyperparameters, a gridsearch is performed which creates a model with all combinations of the hyperparameters and gives the one with the best result.

The decision tree achieved an accuracy of 50% on the testing data which is comparable to the null model.

Random Forest

The random forest is composed of multiple decision trees, hence the name forest. The theory of the Random Forest is that many small learners will combine their knowledge and make better group decisions (Géron, A. 2019). This is an ensemble machine learning model meaning it uses a combination of many different models to make its predictions. Each decision tree will make a prediction and vote for that classification. At the end, the class with the most votes will get the final classification.

The random forest is worse than the decision tree and null model and achieved an accuracy of 45% on the testing data.

Neural Network Model

Finally, a neural network is created. As explained in the Machine Learning Models section, this is one of the most advanced machine learning algorithms. It has the ability to capture complex relationships of variables through its weights and activation functions. A simplified example of the neural network is shown below.

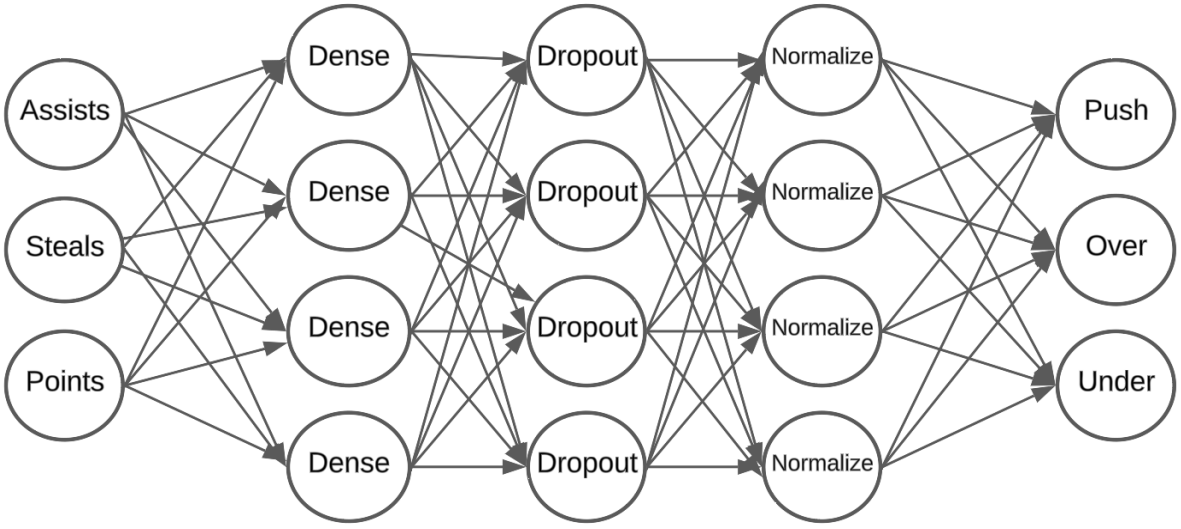

Diagram 2: Simplified version of sports betting neural network.

The neural network model is constructed as a sequential neural network and is as follows:

1. Dense Input Layer

   The input layer consists of 157 nodes, one for each of the variables. ReLu activation function is used.

2. Batch Normalization Layer

   Batch normalization is a technique that improves the accuracy of the model. It recenters the variable and retains the normalization process performed at the beginning (Keras Team).

3. Dense Layer

The dense layer is a fully connected layer consisting of 128 nodes where weights are calculated. ReLu activation function is used.

4. Dropout Layer

   The dropout layer helps the model reduce the chance of overfitting (Keras Team). 20% of the inputs in this layer will be set to 0 which will place importance on other variables.

5. Batch Normalization

   Batch normalization is performed to restore the variables to the same scale.

6. Dense Layer

   The dense layer is a fully connected layer consisting of 64 nodes where weights are calculated. ReLu activation function is used.

7. Dropout Layer

   The dropout layer helps the model reduce the chance of overfitting (Keras Team). 20% of the inputs in this layer will be set to 0 which will place importance on other variables

8. Batch Normalization

   Batch normalization is performed to restore the variables to the same scale.

9. Output Layer

   The output layer is a fully connected layer consisting of 3 nodes where weights are calculated. Softmax activation function is used. This will provide 3 probabilities correlating to the probability each prediction belongs to the corresponding class.

The Adam optimizer is used as the optimization function. This is Keras' adaptive gradient descent algorithm known for its effectiveness in training deep learning models (Géron, A. 2019). Additionally, categorical cross-entropy is employed as the loss function, suitable for multi-class classification tasks such as this one (Keras Team). Since the probabilities of each classification are produced, categorical cross entropy will compare the similarity between the true probability distribution and the probability distribution produced by the model.

Output of the Model

The output of the model consists of three probabilities, each corresponding to the likelihood that the observation belongs to one of the classes: “Over”, “Under”, or “Push”. To make predictions, the class with the highest probability for each observation is predicted. The model’s confidence in its prediction can be judged based on these probabilities. If one is much higher than the others, say 90%, 5%, and 5%, it can be concluded that the model is more confident in its prediction than if the output is 33%, 33%, and 34%.

Results

The neural network is the best-performing model with an accuracy of 53.98% on the testing data. This outperforms the null model and will provide a very slim profit.

The confusion matrix below shows where the model was correct and where it was incorrect. The predicted labels are on the x-axis and the true labels are on the y-axis.

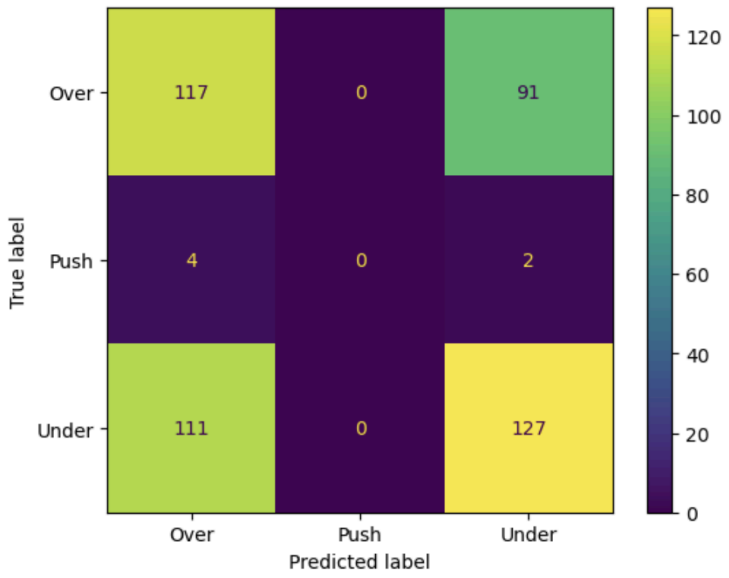

Diagram 3: Confusion matrix for neural network

The takeaway from this diagram is that the model did better predicting “Under” than “Over” and the model never predicted “Push”. This is probably because “Push” only occurred 6 times in the testing data.

Despite achieving a 53.98% accuracy rate, the profitability remains slim. However, a deeper analysis of the model's output reveals potential ways for increasing profitability. By considering the possibility of not placing a bet, the betting strategy can be enhanced. For instance, if the

model assigns probabilities of 33% for “Over”, 33% for “Push”, and 34% for “Under”, indicating low confidence in its prediction, refraining from betting may be the best option. The following graph shows the effects of not placing a bet if the max probability is not high enough. The x-axis represents the model's confidence in its prediction and the y-axis represents the accuracy of the bets.

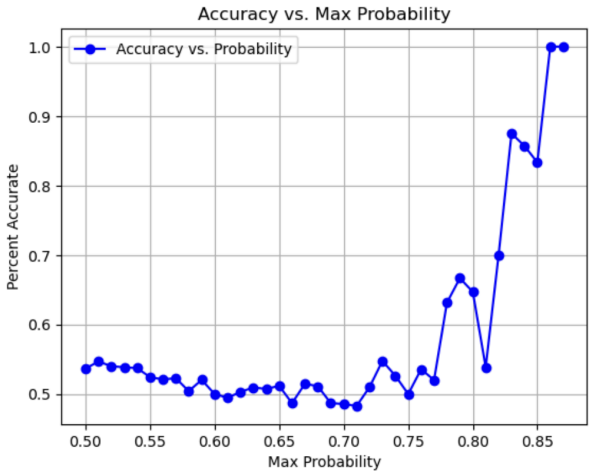

Diagram 4: Accuracy vs max probability threshold

Based on the graph above, it seems like there are good results when only placing a bet if the model is over 85% confident in its prediction. This increases the accuracy to 83% which would greatly improve the profitability. The problem with this is that there were only 6 occurrences of the model being over 85% confident. This greatly reduces the amount of games that could be confidently bet on. There are 452 games in the testing set meaning bets would only be placed on 1% of games.

Moving Forward

To further enhance and advance this project, several steps can be taken:

1. Threshold: A threshold-based approach could be implemented, only placing bets when the maximum probability exceeds a certain threshold. This strategy has the potential to improve the accuracy of the bets and enhance profitability.
2. Data Collection: Expanding the dataset beyond just the previous two games' data could significantly improve prediction accuracy. Incorporating additional data sources such as player lineups, injuries, team performance trends, and other relevant game factors could provide valuable insights for more accurate predictions.
3. Model Complexity: Experimenting with more complex model architectures such as recurrent neural networks (RNNs) or long short-term memory (LSTM) networks could

better capture time-related nuances and improve prediction accuracy, especially when considering the sequential nature of NBA games.

4. Evaluation and Validation: Continuously evaluating and validating the model's performance on unseen data is crucial for assessing its effectiveness and identifying areas for improvement. Implementing rigorous validation techniques such as cross-validation can improve the model's strength.

Ethical Considerations

Several ethical considerations arise when developing machine learning models. Firstly, the collection of accurate data must be carefully scrutinized to ensure it was obtained legally, accurately, and comprehensively. In this project, data was sourced from two places: a free NBA API providing historical game stats and a proprietary API providing historical odds.

Transparency of the model is another critical ethical consideration. While neural networks, like the sequential neural network used in this project, can be complex, efforts need to be made to explain their workings thoroughly. If this machine learning model were to be published, it would be essential to provide users with a clear understanding of how the model operates to prevent any misunderstandings or misinterpretations. There are instances where the stakes of machine learning models can be very high, such as predicting if someone will reenter jail or predicting a cancer diagnosis. Since this specific model is used for recreation and will not be commercialized, this is less important for this situation. The intended user of this model is the creator, therefore any misunderstanding and misinterpretation are self-caused. If this model were to be released publicly, more rigorous testing and reasoning behind decisions would be necessary.

Furthermore, ethical concerns arise from the intended use of the model, which aims to predict sports betting outcomes for the user to improve their odds. Potential issues include addiction to sports betting and misuse of the model to manipulate outcomes. Blindly following the model without considering contextual factors that the model doesn’t account for or wagering amounts could lead to adverse consequences. For machine learning models in general, misrepresentation of the model's accuracy and precision by its creator could mislead users and result in lost bets. Again, this is not an issue for this model since its user is its creator.

Conclusion

In conclusion, this model outperforms the null model by 3.32% and has the potential to generate a modest profit if utilized consistently. However, it is not without its challenges of collecting accurate and relevant data.

By examining the model's output and considering its confidence in predictions, more informed betting decisions can be made and profitability can be increased by refraining from betting when the model is not confident in its prediction. While this strategy may offer the possibility of greater profits, it greatly decreases the quantity of bets made.

To further enhance and sustain this project, additional data collection is needed. This includes expanding the dataset to incorporate more comprehensive information and exploring ways to increase access to historical odds data. Incorporating injuries, game starters, consecutive away games, or other non-play-related data could capture factors related to player well-being and possibly increase accuracy. This would require restructuring the data to accommodate these features by finding a way to accurately represent this qualitative data. By addressing these challenges and continuing to refine the model, the performance of the model can be improved.

References

Bonesteel, M. (2022, August 29). Sports betting timeline: From Las Vegas to the Supreme Court. The Washington Post. Retrieved from https://www.washingtonpost.com/sports/2022/08/29/history-of-sports-gambling/

Bucquet, A., & Sarukkai, V. (2018). The Bank is Open: AI in Sports Gambling. Stanford University. Retrieved from https://cs229.stanford.edu/proj2018/report/3.pdf

CBS Sports Staff, Bengel, C., & McCarriston, S. (2023, November 17). U.S. sports betting: Here is where all 50 states currently stand on legalizing online sports betting sites. Retrieved from https://www.cbssports.com/general/news/u-s-sports-betting-here-is-where-all-50-states-currently-stand-on-legalizing-online-sports-betting-sites/

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (2nd ed.). O'Reilly Media.

Jensen, B. (2022, March 31). Beating the Books: Using Machine Learning to Make Money Sports Betting. CodeX.

Keras Team. BatchNormalization layer. Keras Documentation. Retrieved from https://keras.io/api/layers/normalization_layers/batch_normalization/

Patel, S. (2024). nba_api. GitHub. https://github.com/swar/nba_api