Implementing a Multilayer Perceptron to Identify Market Inefficiencies in NBA Betting Odds

Multilayer Perceptrons for NBA sports betting

Lathan Gregg, uua9gw | Trenton Slocum, nuf8ms | Zach Stautzenbach, ehe5bn

1. Motivation

Since the repeal of the Professional and Sports Protection Act (PASPA) in 2018, which outlawed sports betting for the majority of the United States, Americans have wagered an estimated $466 billion on sporting events. This rapid growth in the sports betting market presents opportunities to analyze and potentially exploit inefficiencies in how betting odds are set.

In addition to accepting bets and paying out winnings, bookmakers set odds for sporting events, which represent the potential payouts a bettor would receive for wagering on a certain outcome of an event. When setting odds, bookmakers aim to maximize their profits. They do this by setting accurate odds, which, in addition to a house margin, balance their risks. However, the odds may not be perfect because bookmakers have imperfect knowledge and might intentionally adjust their odds to balance their risks and take advantage of bettors' biases.

In this project, we examine if a deep learning model is capable of identifying and exploiting inefficiencies in the NBA betting market. We apply the TabNet model to predict which team will cover the spread, the predicted winning margin set by sportsbooks (Vegas), based on team statistics and betting data available at the time of the game. We compare the TabNet performance to predictions generated by a Support Vector Classifier (SVC) and a Random Forest model, evaluating the accuracy, calibration, and return on investment (ROI) over the course of two test seasons.

2. Dataset

Our data was obtained from Kaggle and consists of National Basketball Association (NBA) box score statistics and betting odds from the 2012-2013 through 2018-2019 regular seasons and playoffs (NBA Odds and Scores). Each season has 3 datasets: *raw_scores*, *vegas*, and *vegas_playoff*. The *raw_score* datasets contain scores and statistics for both teams in every regular season game. Each file contains 28 columns with around 2460 entries per season.

The *vegas* dataset records pre-game betting information offered by Vegas sportsbooks. It includes lines and odds for three common wager types:

- **Spread bets**: wagers on the point margin by which one team will win or lose.
- **Moneyline bets**: wagers on which team will win the game outright.
- **Over/under bets**: wagers on whether the total combined points scored by both teams will be over or under a predetermined threshold.

The *vegas* dataset contains 58 columns with 2460 entries for each season with information from five different sports books (Pinnacle, Bovada, Betonline, Heritage, and 5dimes). For this project, we focused on predicting which team will cover the spread. Data pertaining to the moneyline and over/under were used as predictive features.

We did not use the *vegas_playoff* dataset because of incomplete data and difficulties presented by predicting highly variable playoff games. In total, our dataset combines game

outcomes, team statistics, and market expectations for 8610 regular season NBA games from 2012-2019.

3. Related Work

We reviewed multiple papers that implemented machine learning models to predict match outcomes. One paper, written by Sascha Wilkens, looked at sports predictions and betting models in tennis matches. She explained how average prediction accuracy typically can't be higher than 70% when testing multiple types of models. However, the neural network model did achieve an accuracy of 70.4%. Models were also tested on different betting strategies such as fixed amount bets or fixed proportion of the current bankroll with most returns being negative for both options. It is mentioned that recurrent neural networks are a possible solution to this problem in that connections between neurons can form directed cycles which means how a team performs previously could impact the game outcome.

We also reviewed a 2024 paper by Conor Walsh and Alok Joshi that tested whether model calibration is more important than accuracy for sports betting. A highly calibrated model is one in which the probability classifier closely reflects the true frequency of an event. The paper compared a model optimized for expected calibration error (ECE) and a model optimized for accuracy and concluded that optimizing for calibration resulted in a higher ROI compared to accuracy. Walsh and Joshi tested four machine learning models for each objective: a logistic regression model, a support vector classifier (SVC), a random forest model and a MLP. While the SVC was chosen for evaluation based on its performance on the validation set, the MLP performed similarly for both calibration (3.59% ECE vs 3.23% ECE) and accuracy (65.69% vs 66.55%). These results indicate a MLP model optimized for calibration might be capable of exploiting market inefficiencies in the NBA betting market.

4. Technical Approach

In this study, we predicted if the home team would cover the average spread set by Vegas sportsbooks. For this task, we implemented the TabNet model using the pytorch TabNet implementation. We then implemented a simulation to assess the model performance over the course of our two test seasons and compared the results to predictions made by the SVC and Random Forest.

4.1 Data Preprocessing

This dataset required significant preprocessing and feature engineering to fit our models and ensure that only the information that would have been available prior to the game, when the wager would be placed, was included.

First, we calculated our target variable, a binary feature representing if the home team covered the spread. Next, we engineered several predictive features using a combination of team performance statistics and market-based metrics. These included:

- Vegas-implied win and cover probabilities: calculated from the spread and moneyline odds and representing the vegas predicted probability for the home team to win and cover
- Winning percentages: percentage of season games won to date
- Public favorability: computed as the difference in log-odds between the percentage of bets placed and the Vegas implied probability

- Season average statistics: average team points, rebounds, assists, turnovers and shooting percentages

To avoid data leakage, we implemented lagged rolling averages for all season average statistics, ensuring the information for the current game was not available to the model at the time of predictions.

We then dropped metadata columns and other features not relevant for prediction, including team names and ids and raw betting lines. Finally, we standardized all features to have a mean of zero, which was particularly important for our SVC which is sensitive to the magnitude of features.

4.2 TabNet Model

To predict if the home team would cover the spread, we trained a TabNet model on our training data. TabNet is a deep learning architecture proposed by Google Cloud AI researchers Sercan Arik and Tomas Pfister specifically designed for tabular data. The model is structured to efficiently learn feature representations and make human-like decisions through a series of attentively chosen features at each step of its process. The technical details of the TabNet model are reviewed below and the model architecture is shown in Figure 1.

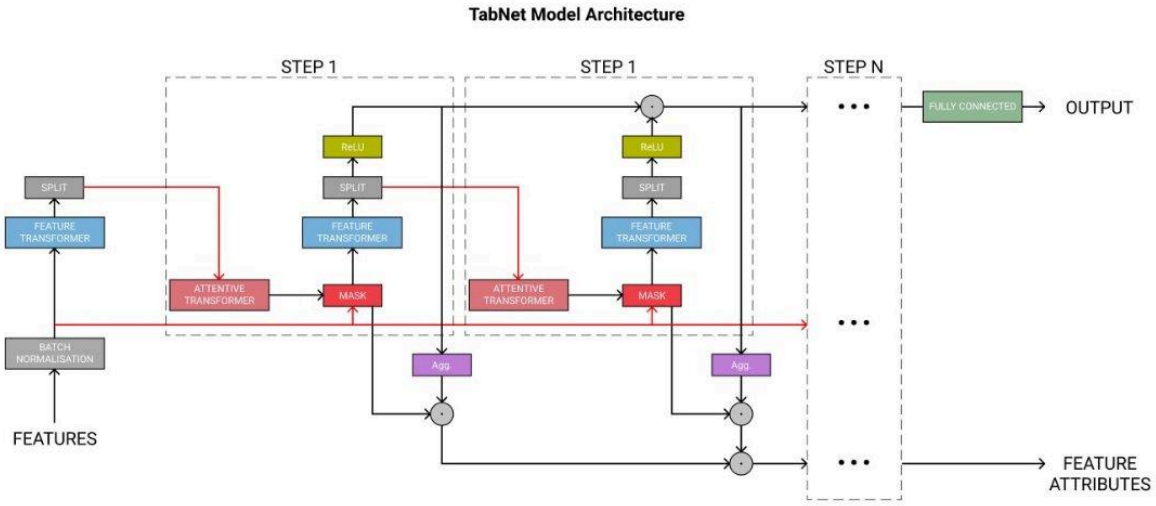

Figure 1 (link)

4.2.1 TabNet Model

The input features first undergo a batch normalization step. This ensures that the data is standardized, providing consistent scales for each feature. By normalizing the features, TabNet can train more efficiently, improving both convergence speed and stability. The normalized data then passes to the feature transformer block. Here, the model begins learning meaningful representations of the input data. The feature transformer captures relationships and patterns that are crucial for decision-making in predicting game outcomes.

This split allows TabNet to focus on different features at different steps, mimicking human-like decision processes. By iteratively updating the attention masks, the model dynamically chooses which features to focus on for better prediction accuracy.

After several steps (as depicted in figure 1), the model aggregates the outputs of each step and passes them through a fully connected layer to produce the final predictions. The final output

reflects the learned relationships between features, enabling the model to provide accurate predictions for game outcomes. Alongside predicted probabilities, TabNet also outputs feature importance scores which allows us to understand which variables impact the response variable the most. This architecture not only improves prediction accuracy but also enhances model interpretability, making it a powerful choice for applications involving complex tabular data.

4.2.2 TabNet Implementation

We implemented TabNet using the PyTorch TabNet package. We set the maximum number of epochs to 200, used a patience of 20 to allow for early stopping, batch size of 512, and virtual batch size of 128 which allows full batch sizes to be split up which helps for regularization. These choices allowed our model to learn efficiently and effectively.

4.3 Comparison to other ML Models

To understand the effectiveness of a deep learning framework, we also trained a Support Vector Classifier (SVC) and a Random Forest model for comparison. The SVC is a supervised learning algorithm that finds a hyperplane to maximize the margin between different classes. SVCs can handle nonlinear data by using kernel functions, but might not perform well with noisy data and can be significantly impacted with the choice of kernel. For our implementation, we used the RBF kernel with a C parameter of 1.0, and set probability to True to enable probability estimates, with a random_state of 42 for reproducibility.

Random Forest is an ensemble method that builds multiple independent decision trees on random subsets of the data and aggregates their predictions through averaging. While this model is robust against overfitting, it can become computationally intensive and may struggle with high-dimensional data as compared to a neural network. In our implementation, we used 200 estimators for the Random Forest model.

5. Experiments

To understand our model's performance, we analyzed the calibration of the model, its accuracy on the test set, and the simulated return on investment over the course of two test seasons.

5.1 Model Performance

The TabNet model ran for 23 epochs but achieved the best log loss of 0.693 in epoch 3. The model trained in 7 seconds using a single 2.3 GHz Intel I9 CPU. The validation log loss over epoch is shown in Figure 2.

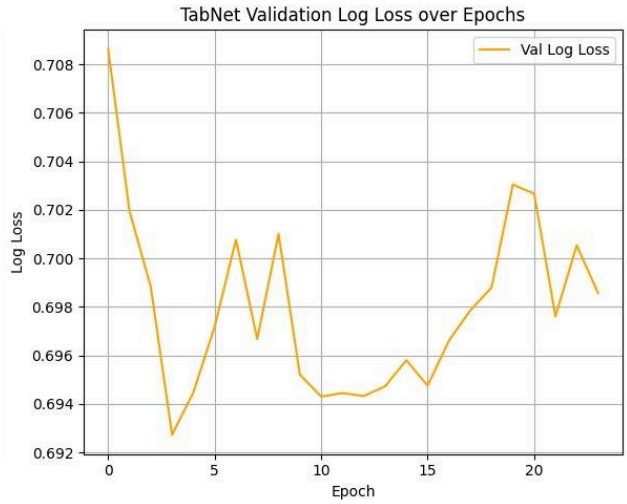

Figure 2

5.2 Evaluation Metrics

We conducted evaluation using multiple different performance metrics. First, we tested our model on the 2017-2018 and 2018-2019 basketball seasons. It achieved a test accuracy of 49.1%. Along with that, we examined the ROC curve shown in Figure 4. This allowed us to see how well it balanced between the true positive rate and false positive rate. In our case, we achieved a test ROC area under the curve (AUC) of 49.0%. Both the test accuracy and ROC AUC tell us that our model is on par with random guessing. While these results may seem low, it's important to note that in betting scenarios, an accuracy of around 52% is often sufficient to break even. Our model approaches that threshold, showing there is potential with further refinement.

In addition to observing the accuracy on the test data with the ROC AUC, we assessed how well the model could be calibrated. This refers to how closely the predicted probabilities match the true likelihood of outcomes. A well calibrated model predicts probabilities that reflect true event frequencies. For example, if the model predicts the probability of the home team covering to be 70%, then 70% of the time that home team should cover the spread. Looking at Figure 5 , the blue line shows the calibration of the TabNet model. It fluctuates a lot, showing low-quality calibration which could lead to consistently over or underestimating predictions along with the possibility of over-allocating bets.

To provide a benchmark, we evaluated the same metrics for a support vector classifier and a random forest model. The test accuracy on the SVC achieved 49.7% while the random forest achieved 48.6%. Similar to the TabNet model, both SVC and random forest are not well-calibrated, as shown in Figure 5. Overall, these models all perform very similarly, having around 50% test accuracy and room for improvement in terms of calibration.

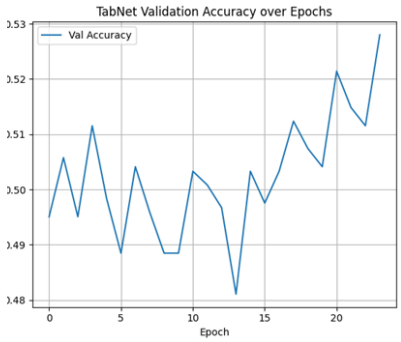

Figure 3

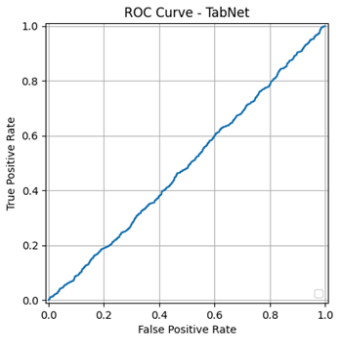

Figure 4

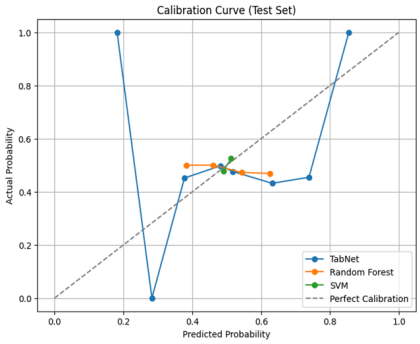

Figure 5

5.3 Feature Importance

To understand what features our model used for its predictions, we analyzed the feature importance scores using TabNets built in feature importance scores which show the percentage of feature selection activity across all decision steps (Figure 6). The most important feature was the number of games played by the home team, which was included in nearly 13% of all feature selection activity. The remaining features were significantly less important, with the 3-point field goal percentage of the away team and the average number of assists by the home team being the only other features used in more than 6% of selection activity.

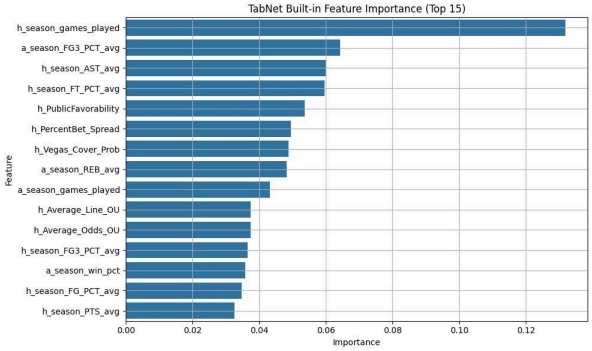

Figure 6

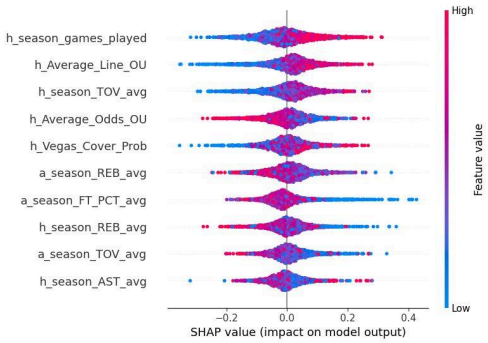

Figure 7

Since TabNet's built in feature importance scores do not give information on the direction of each feature's importance, we also calculate SHappley Additive exPlanation (SHAP) scores (Figure 7). These calculations indicate that the model tended to predict the home team to cover more frequently when they had played more games. The results also indicated that the model favored the home team to cover when the game was projected to be high scoring and the home team had more turnovers.

5.4 Simulation

5.4.1 Implementation

To evaluate how our model performed over the course of our two test seasons, we implemented a simulation using our trained model. We started with a bankroll of $1,000 and bet

on all games in which the probability of the home team covering the spread predicted by our model differed from Vegas's implied probability by over 0.02. To determine our bet size, we tested two strategies. The first was using a fixed bet size of $100. The second used a conservative variation of the Kelly Criterion, which considered the model's confidence and the current bankroll to determine the wager value. The fraction of the current bankroll to wager using this strategy was determined by the formula $f = \frac{1}{8}(bp - q)/(b)$ where:

- f is the fraction of the bankroll to bet
- b is the decimal odds minus 1 (i.e. the net odds)
- p is the probability of winning
- q is the probability of losing $(1 - p)$

We limited the maximum wager to $1,000,000 and ensured winnings from the current day could not be reinvested until the next day.

We ran the simulation with the TabNet model as well as the Random Forest and SVC model. For the SVC model, the Kelly betting strategy resulted in no wagers being placed. For each simulation, we tracked the number of wagers placed, the win rate, and the return on investment.

5.4.2 Results

The results for the simulations are shown in Table 1. The TabNet model resulted in a win rate of 50.1% with both betting strategies indicating that it was hardly more effective than a coin flip in identifying which team would cover the spread. The simulated win rate did increase relative to the test accuracy, indicating that the model was more effective when betting on games with higher confidence. Both strategies resulted in a negative return on investment, indicating the TabNet model was not able to exceed the house edge built into each bet.

The TabNet model had a higher win rate and ROI than the SVC and the Random Forest model simulated with the fixed betting strategy. However, the Random Forest model with the Kelly betting strategy resulted in the highest win rate and ROI. These comparisons indicate the potential of the TabNet model to consistently identify which team will cover the spread. However, it did not provide a clear performance improvement compared to the SVC and Random Forest models.

Model	Strategy	Num. Wagers	Win Rate	ROI
TabNet	Fixed Wager	226	50.1%	-42.32%
TabNet	1/8th Kelly	2102	50.1%	-76.9%
SVC	Fixed	212	49.5%	-83.6%
Random Forest	Fixed	57	42.1%	-100%
Random Forest	1/8th Kelly	626	51.6%	-15.3%

Table 1

6. Conclusions

Overall, building a deep learning model to beat Vegas is a challenging task. Access to vast sources of data and advanced models allow the betting markets to have an advantage with very few exploits. In our case, the additional learning capacity of TabNet was not particularly

beneficial since it resulted in lower accuracy and simulation performance compared to the Random Forest model. However, we did manage to get close to the break-even threshold of around 52%.

Considering our analysis of feature importance and the test accuracy under 50%, it did not appear that the model was able to identify features that Vegas did not accurately account for. Since sportsbooks have access to all of the features in our dataset, they likely already account for the most predictive features when setting the lines.

In terms of future work, incorporating additional features such as player level results and more advanced performance metrics could lead to both better model prediction and calibration. Additional work could also identify specific subsets for which the model accuracy is greater than 50% and strategically bet on these games. Testing alternative loss functions specifically designed to improve calibration may also improve simulation results. Finally, deep learning frameworks designed for time-series data, such as Recurrent Neural Networks, may better account for changes to team performance throughout the season.

References

Eric. NBA Odds and Scores. https://www.kaggle.com/datasets/erichqiu/nba-odds-and-scores, 2020. Kaggle.

S. Arik and T. Pfister, "TabNet: Attentive interpretable tabular learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, pp. 6679–6687, May 2021. doi:10.1609/aaai.v35i8.16826

Shafi, A. (2021, June 7). *TabNet: The End of Gradient Boosting?* Medium; TDS Archive. https://medium.com/data-science/tabnet-e1b979907694

Walsh, C., & Joshi, A. (2024, February 28). *Machine Learning for Sports Betting: Should model selection be based on accuracy or calibration?*. Machine Learning with Applications. https://www.sciencedirect.com/science/article/pii/S266682702400015X#sec8

Wilkens, S. (2021, February 9). *Sports prediction and betting models in the machine learning age: The case of tennis.* Sage Journals. https://journals.sagepub.com/doi/full/10.3233/JSA-200463

Železný, F., Šourek, G., & Hubáček, O. (2019, February 19). *Exploiting sports-betting market using machine learning*. International Journal of Forecasting. https://www.sciencedirect.com/science/article/pii/S016920701930007X?casa_token=rN1QmS2KVlUAAAAA%3AGfRB1qtRfUC0OL85U61ekIHzHw9k0sgchVZahhtxZ0vaulZy_NV9NP6mtmxB9uVQoMTWsnL-