Optimal Strategies for MLB Game Outcome Wagers

César Dori†, Brandon Hadfield‡

1. Motivation and Introduction

Sports analytics is a multi-billion-dollar industry driven by advances in data availability and modeling capabilities [1]. Given the speed with which this field has emerged, it seems likely that heuristic methods are widely adopted. This makes Major League Baseball a good test for the analytical modelling techniques learned in class: games being played every day that are well-documented and supported by rich datasets such as Statcast, Retrosheet, and sportsbook lines. That being said, forecasting MLB game outcomes is a fundamentally challenging task because the sport is inherently high-variance with small margins, influenced by many contextual factors such as pitching lineups, ballpark conditions, and randomness in individual game events [2].

It is also important to realise how stacked against us the odds are in sports betting. Sportsbooks attempt to incorporate all of this information into betting odds, producing highly refined market-based predictions. Because those odds are defined based on analyst expertise, public information, and highly researched bettor behaviour, it is extremely difficult to outperform them. This makes sports betting an interesting benchmark for prescriptive analytics: it is a noisy environment with a very clear prescription (to place a wager or not) and a measurable reward (profit or loss).

Our goal in this project is to experiment with different modelling methods that use MLB data to prescribe wagering decisions. Motivated by the methods covered in class, we explore four complementary modelling pipelines:

1. **Direct IAI Optimal Policy Trees**: The method learns a policy to bet or not from historical rewards. It does not use an explicit win-probability model.

2. **Alternative Rewards Estimation with Policy Trees**: We first estimate rewards using predictive models to predict win probabilities. We then fit an Optimal Policy Tree on the estimated rewards.

3. **Optimal Prescriptive Trees for Direct Policy Learning**: We apply optimal prescriptive trees that learn both the structure of the tree and the wagering policy in one optimisation.

4. **Heuristic Methods**: Here, we construct simple benchmark policies, like always bet the favourite, to provide a baseline for evaluating the value added by our prescriptive models.

A key motivation for the project is that interpretability is important. Modern black-box models can achieve high predictive accuracy, but they offer limited transparency on why a bet is recommended. Optimal Classification Trees (OCT) and Optimal Policy Trees (OPT) allow the decision-making to be inspected, validated, and communicated.

† MBAn Student, MIT Sloan Management School.
‡ Graduate Student, MIT Deptartment of Aeronautics and Astronautics.

This aligns with the core theme of the class, which combines predictive modelling with prescriptive optimisation while retaining interpretability.

2. Data Collection and Preprocessing

We combined three complementary baseball datasets to construct game-level features for both prediction and prescription. Each dataset provides information at a different level of granularity: historical game outcomes, team performance in each game, and starting-pitcher performance in each game. Although these rich resources provide data for free, for many years, the dataset collected ranged from the 2019 - 2025 seasons.

Table 1 summarises the key components.

Dataset	Source	Granularity	Contents
Retrosheet game logs	Retrosheet.org	One row per team–game	Game dates, home/away teams, final scores, and starting–pitcher identifiers.
Team Statcast game stats	Baseball Savant	One row per team–game	Team-level batting and contact metrics (e.g., wOBA, xwOBA, hard-hit%, barrels).
Pitcher Statcast game stats	Baseball Savant	One row per pitcher–game	Starting–pitcher velocity, xwOBA allowed, strikeout and walk rates, and pitch characteristics.

Table 1. Summary of datasets used in model construction.

In addition to these baseball datasets, we scraped moneyline odds from an online sportsbook for the MLB regular seasons since sports betting was legalized in the US from 2019. The odds data provide home and away prices for each game. They are used only in the prescriptive step to compute realised and expected betting returns and not as features for the predictive models.

2.1. Merging Datasets

After collecting the three datasets in Table 1, we combined them into a single game-level table that would serve as our complete representation of each game on which to train our predictive model. The preprocessing steps taken were the following:

1. **Aligning Team Identifiers**: Retrosheet game logs and Statcast datasets use different names for teams. To counter this, we standardised all team IDs to MLB's canonical three-letter codes to join across datasets.

2. **Linking starting pitchers across datasets**: Retrosheet uses Retrosheet-format pitcher IDs (e.g., glast001), whereas Statcast uses MLBAM IDs. We had to build a lookup table mapping Retrosheet IDs to MLBAM IDs to ensure that each game was matched to the correct starting pitcher's Statcast game-level statistics.

3. **Collapsing team-level rows into a single game row**: Retrosheet and Statcast team datasets contain one row per team–game so we merged both datasets using the shared game_pk to produce one consolidated row per game.

4. **Adding game-odds to consolidated dataset**: Finally, we linked odds by joining on game date, home team, away team, after standardising team names/IDs.

The final dataset contains a complete representation of each game, combining team-level performance, starting-pitcher characteristics, game odds, and the true game outcome.

2.2. Feature Engineering

The merged dataset contains hundreds of raw variables, many of which are either noisy, redundant, or unusable for predicting game outcomes. We therefore engineered a focused set of features that capture meaningful aspects of team form, pitcher performance, and game context. All features were constructed using only information available prior to each game to avoid data leakage. Our feature set is organised into four groups:

1. **Rolling Team Performance**
    - Win percentage
    - Average runs scored
    - Average runs allowed
    - Average run differential
    - Rolling offensive xwOBA
    - Rolling offensive launch speed
    - Rolling offensive hard hit percent

    These rolling averages highlight short-term form and momentum of the teams. MLB teams play very frequently, so performance over the last few games is often a strong signal of how good they will do!

2. **Season-to-Date Team Strength**
    - Cumulative win percentage
    - Home-only win percentage
    - Away-only win percentage

    These features summarise teams' performances in the current season. Since rolling averages can be noisy, these features provide a more stable and long-term estimate of the team quality.

3. **Starting Pitcher Performance**
    - Average fastball velocity
    - Rolling weighted on-base average allowed
    - Rolling expected weighted on-base average allowed
    - Rolling launch speed allowed
    - Rolling hard hit percentage allowed
    - Rolling strikeout rate (K%)

- Rolling walk rate (BB%)

The starting pitcher is one of the strongest determinants of game outcomes. Rolling pitcher metrics capture recent form without leaking information from the current game.

4. **Contextual Factors**
   - Home and away park factors
   - Days since each team's previous game

   Although less influential than performance-based features, these contextual variables help explain situational effects such as fatigue and run environments.

The features engineered are the strongest predictors used in MLB game predictions in the industry, supported by our knowledge of baseball. Altogether, they provide a good representation of each game, enabling our models to predict win probabilities and prescribe betting decisions reliably.

3. Modeling Methods

As previously discussed, one of the aims of this project was to use many of the modelling techniques discussed in class to facilitate comparisons both on an objective profit level as well as relative performance. Therefore, an objective baseline strategy is necessary, as well as a standardised way to ensure betting strategies are comparable. Furthermore, in order to ensure there would be no data leakage, policies were trained on seasons prior to and including 2023, then evaluated on 2024 and 2025. Not only does this avoid historical data influencing future results, it also replicates a realistic scenario where we would be predicting on future outcomes using all available information to that date.

3.1. Baseline Strategy

To facilitate relative performance comparisons, we must know what a naive agent could hope to achieve by applying a simple betting strategy. In order to make sure that this is a strategy accessible to everyone, it must be deterministic, applicable to all games, and not require significant analysis.

These requirements are fulfilled by simply betting on the favourite for every game. Since the favourite is determined by the published moneyline, this is a piece of information that any betting agent would know. Furthermore, since the favourite is expected to win, it is not unreasonable that a naive agent would see this baseline strategy as realistic.

Arbitrarily, it was decided that this naive agent would bet $20 on every game since returns would be computed as a percentage of wagered capital.

3.2. Optimized Strategies

Multiple course strategies were applied to make *optimal* bets. They are summarised in Fig. 1 and discussed in the subsequent subsections. Since this is inherently a predictive-prescriptive problem, they are generally variants on a predictive model for win prediction, and a prescriptive model for assigning bets under a given win outlook.

To ensure fair comparison, each one was trained to decide whether to abstain from betting, or bet \$20 on the more profitable team. This both facilitated the use of optimal prescriptive trees, which require a discrete set of candidates, as well as ensured that any comparisons to the baseline would be affected only by each model's bet-no-bet decisions.

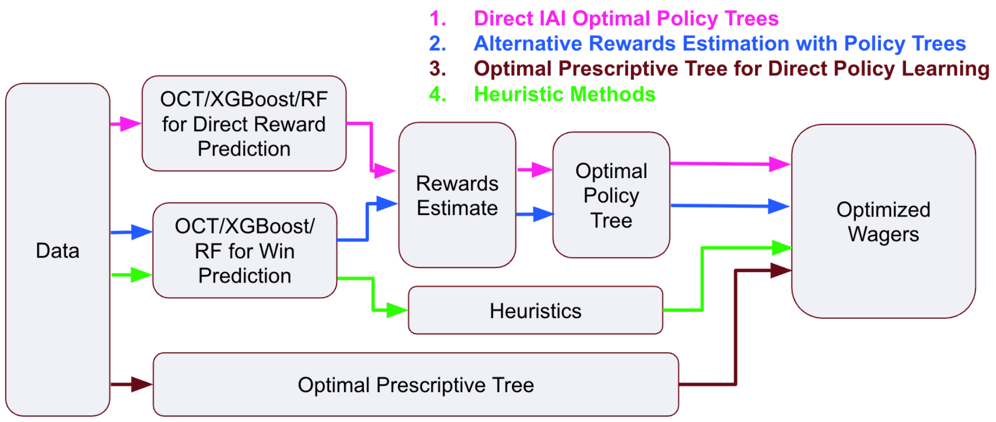

1. Direct IAI Optimal Policy Trees
2. Alternative Rewards Estimation with Policy Trees
3. Optimal Prescriptive Tree for Direct Policy Learning
4. Heuristic Methods
Data
OCT/XGBoost/RF for Direct Reward Prediction
OCT/XGBoost/ RF for Win Prediction
Rewards Estimate
Optimal Policy Tree
Heuristics
Optimal Prescriptive Tree
Optimized Wagers

Figure 1. Visual description of modelling options applied

3.2.1. Optimal Policy Trees

The first model designed was an optimal policy tree using the direct, out-of-the-box implementation provided by `IAI.OptimalTreePolicyMaximizer` and `IAI.NumericRegressionRewardEstimator` [3]. The dataset collected did not include past bets, but the mapping between bets and rewards is a deterministic function, so the dataset was augmented to include a row for each potential bet on each game. That is, for `Game i` there were 3 rows in the augmented dataset:

1. Place a \$20 bet on the home team for `Game i`;
2. Place a \$20 bet on the away team for `Game i`; and
3. abstain from betting on `Game i`.

Under this strategy, the treatment effect is **very** easy to simulate, so the dataset was augmented to ensure that there was no treatment bias. Therefore, it allows the use of the direct method for outcome estimation with the rewards estimation model, which provided the best performance. The ultimate policy tree was trained on a grid search over the `max_depth` and `minbucket` parameters, with the following ranges:

$$\texttt{max\_depth} = [2, 4, 6] \qquad \texttt{minbucket} = [20, 50, 100, 200]$$

3.2.2. Optimal Prescriptive Trees

The second model was an optimal prescriptive tree using the direct, out-of-the-box implementation provided by `IAI.OptimalTreePrescriptionMaximizer` [3]. Since these trees predict rewards during training and in our problem formulation, rewards are a known function of the bet, odds, and outcome it is likely that this style of prescription will lose some performance due to the fact that it has to both learn the form of this function as well as the uncertainty in the outcome distribution.

Again, once defined the tree was tuned on the `max_depth` and `minbucket` parameters, on the ranges:

$$\texttt{max\_depth} = [2, 4, 6] \qquad \texttt{minbucket} = [20, 50, 100, 200]$$

3.2.3. Optimal Policy Trees with Manual Reward Estimation

Since the rewards from a bet are deterministic once the outcome of a game is known, all the uncertainty that a model needs to learn is contained in the game outcome. This inspired the approach to train a win prediction model separately and tune for performance, then use this model to predict the rewards that would be used to ultimately train a policy tree. Here, multiple models were attempted, such as:

- Optimal Classification Trees
- *Random forest style* logistic regression; and
- XGBoost

where the logistic regression model consisted of training a group of logistic regression models, each on a different bootstrapped sample of the training set. The final rewards were then computed by taking a conservative lower bound on the mean response.

3.2.4. Heuristic Methods

As mentioned, all of the uncertainty in the problem is inherent to the outcome of the game, so similar to the previous section, we utilise the best performing game prediction model and apply a set of heuristics to the outcome probabilities to determine when and how to bet. Specifically, a bet would be placed on one of the teams only when the win probability model places at least a 89% confidence on a win.

4. Results

Results were evaluated on one main metric, return on total wager. That is because the absolute profits or losses can directly be manipulated by wagering more on each bet. However, by increasing the amount wagered you also leave yourself exposed to larger losses. Therefore, a return metric computed as

$$Return = \frac{Profit}{Total\ Wager}$$

gives a better representation of the value a model can extract from the wagers it chooses to make. However, there is no doubt that as an individual agent, you may be interested in your total exposure, for which the total profit/loss has also been included. When evaluated on return the models have the performance outlined in Table 2.

Win/Reward Model	**Prescription Model**	**2024 Return**	**2025 Return**	**Avg Return**	**Total Profit/Loss**
Baseline	Baseline	-3.8%	-8.8%	-6.3%	-$5275.09
Prescriptive Tree	Prescriptive Tree	-8.6%	-3.0%	-5.8%	-$4853.07
XGBoost	Policy Tree	**-2.4%**	-9.6%	-5.9%	-$3399.78
Random Forest	Policy Tree	-9.3%	-4.5%	-6.6%	-$1028.18
Logistic Regression	Policy Tree	-16.0%	9.3%	-3.0%	-$174.46
XGBoost	Heuristic	-22.9%	-7.0%	-15.2%	-$509.51
OCT	Heuristic*	-7.1%	**15.9%**	**4.0%**	**$23.35**

*For the OCT-Heuristic model, a probability cutoff of 80% had to be used since there were no games which produced a win probability $\geq 89\%$

Table 2. Return results for the 2024, 2025 seasons. Using an optimal policy tree with XGBoost for rewards regression resulted in the smallest percentage loss on the 2024 season, while the OCT-Heuristic combination provided the best return on all other metrics

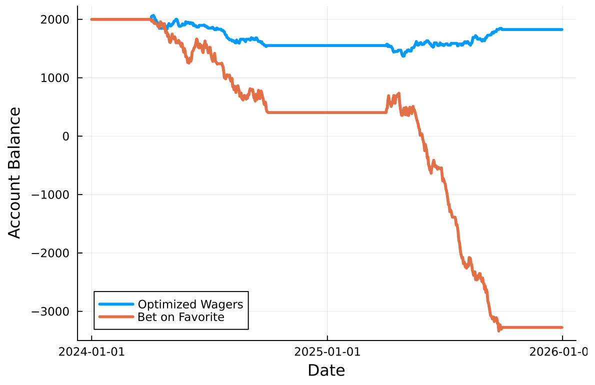
It is immediately clear that the old adage - *the house always wins* - rings true for sports betting. Across the board the optimised methods fail to provide winning strategies, although most do provide some relief from the losses that you would incur if you had simply bet naively on the favourite. In fact, the OCT-Heuristic combination does even yield a winning season, although much of its positive return is owed to its extreme conservatism.

The development of the profit/loss can also be visualised through time to identify when these models do well. This can be seen in Figure 2. Namely, it is immediately clear that the OCT-Heuristic pipeline, which had the greatest return by far on 2025 and even netted a positive overall return, bets so infrequently that on a two-year scale, the changes in balance are nearly imperceptible. This would essentially suggest that the OCT is identifying a subset on which it does perform well; however, this subset is exceedingly small and likely too small to make this a truly profitable betting strategy.

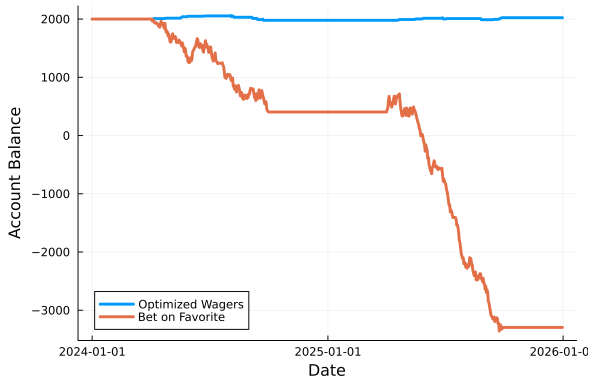
On the other hand, the XGBoost - Optimal policy tree pipeline had a terrible 2025 season, but actually provides the best return on the 2024 season by only losing 2.4%. This is in stark contrast to the OCT model, as it does bet consistently, meaning that its performance is a result of better wagers. Looking at the date breakdown, there are no obvious patterns that correlate with when it does well, although there is an interesting trend approximately halfway through the 2024 season where the favorites are doing poorly, leading to a steep drop in the red curve, while the XGBoost Policy Tree does quite well. This may suggest that there was something around June which led to the model picking up on higher win probabilities which didn't correlate with the moneyline. Investigating this time period may be an interesting case study to improve future performance.

Finally, the logistic regression-policy tree pipeline strikes a medium balance and turns a non-negligible profit on the 2024 season. It places enough bets that the changes in balance are visible at this scale, and interestingly, it has its best performance on the 2025 season, which both the XGBoost and baseline strategies struggle with greatly. Similar to the previous discussion, this may suggest that in the 2025 season there was something that this model was particularly picking up that was not reflected in the moneyline.

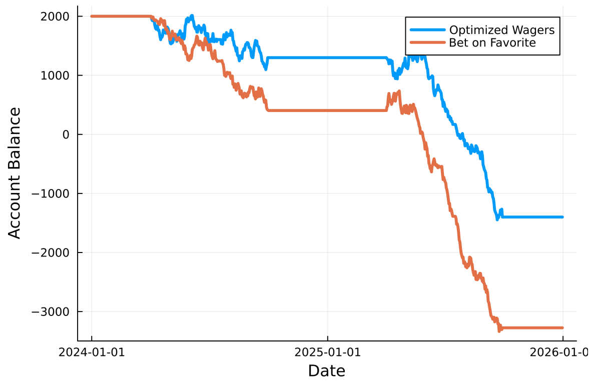
In general, all models show that although there is no glaringly obvious pattern for when these models can perform



(a) Two season results for Logistic Regression - Optimal Policy Tree pipeline



(b) Two season results for OCT - Heuristic pipeline



(c) Two season results for XGBoost - Optimal Policy Tree Pipeline

Figure 2. Two season results visualized for highest performing models

well, the most important factor in producing good wagers is not only having good predictions of game winners, but also knowing when your predictions are in disagreement with the published moneylines. That is to say, you are not only trying to make good predictions of the outcome, but you are trying to understand when your prediction of the outcome is outside of the range implicitly suggested by the bookie's odds.

5. Conclusions

Sports betting is a major industry, but also a highly risky and uncertain one. Our study has shown that although optimisation offers the opportunity to avoid the major losses that one can expect by naively betting, it does not offer a way to make any significant return on your wagers. Specifically, it was found that the best models are able to place wagers when they have a belief surrounding the outcome of the game, which is in disagreement with the published odds; however, none of the models tested were able to identify those opportunities consistently enough to turn a meaningful profit. The best model we found was a highly conservative heuristic model, which turned an overall profit of \$23.35, or 4%; however, other win prediction and policy tree combinations yielded more liberal policies with lower overall returns, but potential patterns to be investigated in when they earn a return.

Distribution of Work

The distribution of work for this project was as even as possible, both partners contributed on all tasks. César contributed slightly more on the data collection and preparation, while Brandon contributed slightly more to the modeling and results.

References

[1] G. V. Research, “Sports betting market, 2025 - 2030,” tech. rep., Grand View Research, 2025.

[2] N. Paine, “The imperfect pursuit of a perfect baseball forecast,” *FiveThirtyEight*, 2014.

[3] L. Interpretable AI, “Interpretable ai documentation,” 2025.