Predicting the Major League Baseball Season

Randy Jia, Chris Wong, and David Zeng

***Abstract*—This paper attempts to predict the outcome of games from the 2012 Major League Baseball season. Sporting events are very important to many people, and professional leagues are worth billions of dollars. Baseball, in particular, is not only one of the most popular sports in the United States, but the vast amount of recorded data and statistics made publicly available also lends itself well to machine learning. Realizing that baseball games are quite noisy, in our prediction, we hope to predict games with sufficiently high accuracy and to unveil information about what makes a winning baseball team. A feature set was carefully chosen, and both classification and regression techniques were implemented. The performance of the algorithms were tested on a recent season and the results showed a small degree of success, but also confirmed the suspicion that baseball games are very hard to predict based off statistics alone.**

I. Introduction

SPORTING events are deeply integrated in many people's lives. In the United States, baseball is one of the most popular sports. The highest level of play occurs in Major League Baseball (MLB), a professional league worth billions of dollars. From television contracts alone, MLB earns $1.5 billion each year from domestic viewers [1]. It is important for a baseball team to win games because it will attract higher fan attendance, television viewership, and, perhaps most importantly, revenue.

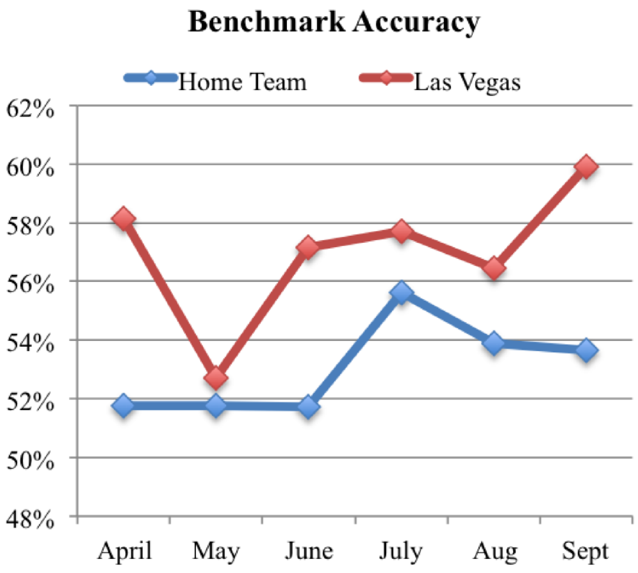
Predicting the outcome of baseball games is an even more lucrative field. CNBC estimates that over $30-$40 billion is wagered on games every year [1]. Luckily, MLB games lend themselves well to machine learning because of the vast amount of data that is recorded for each game. A downside, though, is that baseball itself is quite noisy and thus difficult to predict with high accuracy. Any player having a "good" day can hit a home run, and this action alone can allow for a lesser-skilled team to defeat a better team. After all, if it were that easy to predict the winning team for each game, then why even play the games at all? Nevertheless, we will explore machine learning algorithms using data from recent MLB seasons and try to predict the 2012 season as accurately as possible.

II. Benchmarks

Before we collected, tested, or trained any data, we established benchmarks so that we would be able to measure and analyze any results. We defined two target prediction rates based on: 1) predicting the home team wins every game, and 2) using the predicted winner according to betting odds and spreads from Las Vegas sports books.

Month	Home Team	Las Vegas
Apr	51.78%	*58.13%*
May	51.76%	*52.71%*
Jun	51.73%	*57.18%*
Jul	55.61%	*57.70%*
Aug	53.88%	*56.47%*
Sep	53.66%	*59.90%*
Avg	53.29%	*56.87%*

TABLE I. Benchmark Prediction Accuracies



Las Vegas betting and prediction data was taken from SBRforum [2], a website which aggregates spreads and odds from many popular sports books. Data files were not readily available, so we scraped and interpreted raw HTML from many webpages. For our work, we only used the betting information that "predicted" which team was going to win. We did not use more detailed data such as run spreads or money lines, which we will discuss later as possibilities for future work.

If our results better these benchmarks, especially the percentages from Las Vegas sports books, then our machine learning processes could potentially have tangible and monetary significance. Throughout this paper, we will *italicize* the Las Vegas benchmark percentages and will **bold** any results that surpass them.

III. Machine Learning Approach

The simplest way to predict games is to implement a learning algorithm that classifies games as either a "win" or a "loss." This approach can be achieved using popular machine learning algorithms such as logistic regression, SVMs, logistic boost, and adaptive boost. The classification approach, however, does not indicate the nature in which a game was won or loss; it does not give any information regarding how many runs were scored or by how much a team won. Therefore, we also implement a regression approach, which predicts how many runs are scored by each team. Hence, we can determine who won by a simple comparison of runs scored.

IV. Feature Selection

Because of the sheer quantity of different numbers that baseball statisticians record, it is important to identify which features are the most important in deciding victories. To do this, we must determine which features have the greatest impact on the number of runs scored, thereby determining the winner and loser of a game. We evaluated many of the popular statistics used by baseball statisticians and narrowed down a feature set we believed correlates most to scoring runs:

Batting

- *Batting Average (BA):* number of hits divided by number of at-bats
- *Slugging Percentage (SLG):* number of bases divided by number of at-bats
- *Runs Batted In (RBI):* number of runs a batter's team scores by virtue of a hit
- *On-Base Percentage (OBP):* how often a batter reaches a base per at-bat
- *Walks Drawn (WB):* number of times a batter reaches a base on balls

Pitching

- *Earned Run Average (ERA):* earned runs per 9 innings pitched
- *Hits Allowed (H):* hits allowed to batters
- *Bases Allowed (B):* bases allowed to batters
- *Strikeouts (K):* strikeouts per inning pitched
- *Walks Thrown (WP):* number of times a pitcher allowed a base on balls

Team Stats

- *Errors (E):* number of fielding errors committed by team
- *Past Head to Head Matchups:* record of past games between the two teams
- *Left on Base:* number of batters that reach a base but never score a run
- *Win Percentage (Win%):* current season win percentage

Data was readily available for us to download for free from Retrosheet [3]. The files contained detailed pitch-by-pitch logs of all MLB games for each completed season, with records going as far back as the late 1800s. To ensure the completeness and integrity of the data, we decided to consider only games from 2007 to 2012. The first step of our project was converting the CSV play-by-play data files from Retrosheet into much more accessible forms, such as an sqlite3 database. Data from the years 2007 to 2012 produced about 2 million rows of play-by-play data from which we aggregated box score summaries for each game. We wrote Java programs to query our database and calculate game-by-game data for the statistics listed above. This in turn allowed us to build new CSV files that were much more useful for our feature selection process. There are over 2400 MLB games per season, and so the large sizes made this process very challenging.

Using our box score summaries of each game, our goal was to find which subset of statistics were most indicative of producing a win. From this vast feature set, we utilized feature selection to select the ideal features of choice. First, we ran a best subset feature selection on the features in each of the three categories: batting, pitching, and team stats. However, best subset can sometimes overfit the data, so we also decided to run both forward and backward selection to account for this possibility. The following table displays the results of our feature selection:

Selection Algorithm	Features Considered	Most Important Features
Best Subset	Batting	RBI, BA
Best Subset	Pitching	ERA, H
Best Subset	Team Statistics	E, Win%
Forward	All	E, RBI, BA, ERA, H
Backward	All	E, RBI, OBP, ERA, Win%

TABLE II. Feature Selection Algorithms

From our feature selection analysis, we decided on BA, RBI, OBP, ERA, H, E, and Win% for each team. Preliminary tests indicated that these are the more important features, and adding other features did nothing to improve the prediction.

V. Prediction Model

Along with these features, we must also consider the time frame from which we derive our baseball statistics (e.g. the current season, last 3 seasons, a player's entire career). While the easiest approach would be to just use a player's career statistics, we believe that statistics from the current season (2012) are much more significant than those from previous seasons. In other words, previous season performance is not the best indication of current performance. Factors such as injuries, suspensions, trades, experience, improvement, and health issues can create high variance in each season's performance for a significant number of players. Therefore, we decide to put most of the weight on performances from the current season, and much less weight on past performances. The drawback with this approach is that predicting games from the first 2 weeks of the season will prove to be difficult because of the small sample size of current season data. We will address prediction accuracies for different portions of the season later.

Thus, we use the 2012 season as our test data and the previous five seasons as our training data. The hope is that we can predict the 2012 games with accuracy as well as baseball games can be predicted. The challenge lies in the fact that baseball games are inherently very noisy, and the "better team" is not always guaranteed to win, so it is unreasonable to expect a very low error rate.

To predict the winner of a game, we consider the players that comprise the starting lineup of both teams: 9 batters and 1 pitcher. To predict game N of a season, we accumulate the statistics for the past $N - 1$ games of the season for each player on the starting lineup. These cumulative statistics are the particular game's features. Our machine learning algorithms will then try to determine how the differences in the features between the two teams indicated a win or a loss.

VI. Classification Algorithms

We first trained a logistic regression model and SVM on the 2007-2011 seasons and classified the 2012 season. The SVM was implemented with a Gaussian radial basis function (RBF) kernel and a cost parameter, as the data is most likely nonseparable. The value of this cost parameter was determined by picking the best parameter value from a set of predetermined values ranging from 0.01 to 20 using a validation set approach.

We also decided to try a boosting approach. With classification trees as weak learners, we performed boosting in the form of AdaBoost (adaptive boosting) and LogitBoost (logistic boosting). Boosting utilizes a number of weak classifiers and creates a single strong classifier. As such, boosting reduces bias because of the large number of weak classifiers. Here, the number of iterations controls how many weak classifiers there are. If this parameter is too small, however, the data may be underfit. On the other hand, if it is too large, the data may be overfit. As a result, we perform 10-fold cross validation beforehand on our training set to obtain an optimal number of iterations.

Algorithm	Accuracy
Benchmark	*56.9%*
Logistic Regression	55.7%
SVM, RBF kernel, Cost = 5	**59.6%**
AdaBoost, 35 iterations	**58.5%**
LogitBoost, 25 iterations	**59.5%**

TABLE III. Results of Classification Algorithms

We see that SVM, AdaBoost, and LogitBoost achieve an accuracy of almost 60%, indicating that baseball is a rather difficult sport to predict. We have, though, cleared our benchmark with three of the four algorithms. To see if better results can be obtained, we also try a regression approach.

VII. Regression Algorithms

We fit both a linear regression model and a random forest model. For the random forest model, the number of predictors to be considered at each split in a tree was determined via a validation set. In each model, we predict the runs scored for each team in each game, reporting both the test mean squared error (MSE) for run differential and the classification accuracy after comparing the two teams' scores.

Algorithm	Test MSE	Classification Accuracy
Benchmark	-	*56.9%*
Multiple Linear Regression	16.29	**57.1%**
Random Forest, 7 predictors	13.14	**59.0%**

TABLE IV. Results of Regression Algorithms

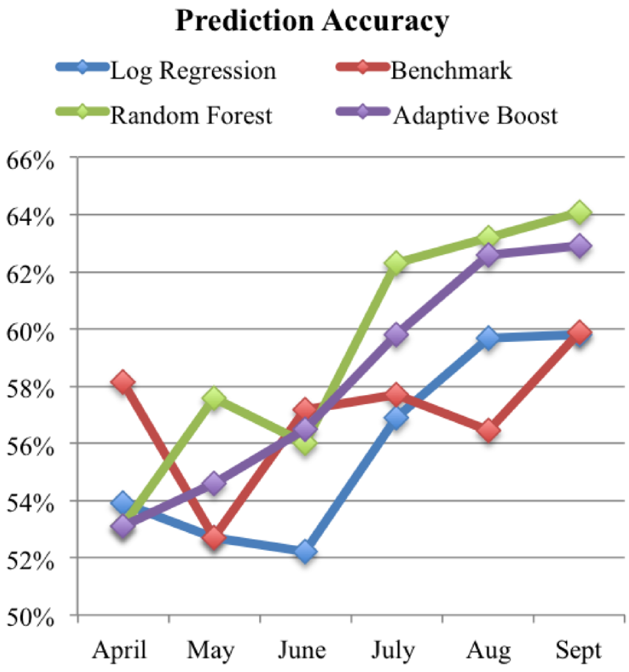
We see that while the classification accuracy was competitive with the classification methods described earlier, the MSE was substantially large, leaving us to conclude that regression is, overall, not an accurate path to consider. In both our regression algorithms, however, we surpassed our benchmark.

VIII. Time-Based Predictions

To uncover the issue of different parts of the season being harder to predict, we decide to partition our test data into months (April to September) for each algorithm. Dividing the test data set in this manner allowed us to confirm our belief that games earlier in the season are harder to predict. The following shows that this is indeed true.

Month	Bench-mark	Logistic Reg.	Adaptive Boost	Random Forest
Apr	*58.1%*	53.9%	53.1%	53.1%
May	*52.7%*	52.7%	**54.6%**	**57.6%**
Jun	*57.2%*	52.2%	56.5%	56.0%
Jul	*57.7%*	56.9%	**59.8%**	**62.3%**
Aug	*56.5%*	**59.7%**	**62.6%**	**63.2%**
Sep	*59.9%*	59.8%	**62.9%**	**64.1%**

TABLE V. Month by Month Accuracy of Various Learning Algorithms



IX. Discussion

The accuracy achieved is not what one would hope for a prediction if, say, one were to bet on a game. Our prediction accuracies were only slightly better than the first heuristic benchmark that naïvely picks the home team to win every time. This supports the hypothesis that baseball is a very noisy sport and difficult to predict. However, we believe we can continue to do better. We believe one major cause for poor accuracy is that the early part of the season is more difficult to predict; there have not been enough completed games to determine how well a player is likely perform for the rest of the season. It is also possible that players are not in shape at the beginning of the season and their performance deviates greatly from their true skill level.

Regardless of the reason for inaccuracy, the small amount of highly variant statistics for early-season games results in our prediction rates being relatively low. Many algorithms fail to meet the Las Vegas benchmark for the first three months by wide margins. This suggests that real-world knowledge and intangible factors (such as a player being in shape) may have a greater significance on these first few games. Within our work, we did not have statistics to measure these details, but oddsmakers can certainly take these into account.

For games in the later half of the season, our results are nearly 10% better than those from the earlier half. Thus, we can see that it is likely that baseball games have some degree of predictability. The cumulative season data that we use as features for each game stabilizes as each player and team adjusts to the season. Indeed, many of our algorithms surpass our Las Vegas benchmark, implying that it is possible to use statistics as one of the better predictors of the game.

Throughout our results, we see that our machine learning algorithms have outperformed a significant benchmark: the Las Vegas oddsmakers for the 2012 MLB season. Note, however, that this does not necessarily mean that we can win large amounts of money by betting based off of statistics. For one, we have only tested our algorithms on one season, and each season as a whole can have intangible variabilities that may or may not apply for any particular one. Additionally, our interpretation of the Las Vegas benchmark is a highly simplified model of how betting actually works. Often times, one cannot just bet on the winner of a baseball game; usually, one must bet on a run spread or money line. The intricacies of betting are outside the scope of this paper, but it should be realized that a greater number of predicted games may not directly lead to gambling winnings. Of course, though, it is a good start.

X. Other Attempts at Improvements

Upon analyzing our learning methods, we realized that our feature set still seemed to be nonideal, either from improper selection or simply that baseball statistics do not provide a good prediction for the outcome of games. To address this, we attempted to include polynomial and interaction terms of the features, but many different combinations of these terms did not help the prediction accuracy at all.

We also ran Principal Component Analysis on the feature space and limited ourselves to varying numbers of principal components; however, this did not improve accuracy (in fact, the performance was actually worse). Running SVM with different kernels, such as a polynomial kernel, also did not improve results.

An indication that our learning algorithms were not performing as well as we desired was the values of the training error. In most cases, the training error matched the test error; this indicates possible underfitting of the data. However, any attempts to remedy this (such as tweaking parameters) failed to improve accuracy. While we still hope to do better, the noisiness of the data and the unpredictability of baseball itself are major roadblocks in achieving this.

XI. Future Work

Despite our results, we cannot say for certain that the upper bound for predictions rates of baseball games is 60-65%; we have not considered all feature sets and learning algorithms. This work can be built upon, and hopefully improved, by possibly expanding the feature set or creating a baseball statistic more indicative to winning baseball games. For example, one could analyze the data at a more granular level to include a specific batter's record against a specific pitcher. One might also consider the difference in playing style from the National League and the American League. Perhaps one might even consider quantifying intangible factors such as how experts feel about a particular game or how physically ready a team's players are. It may also be interesting to try another learning algorithm such as neural networks.

We see area for future work in accurately predicting the playoffs; since these games occur after the regular season, our results suggest we should be able to predict them with higher accuracy. The playoffs are an important time of the season since it is when the baseball champion is crowned, a distinction driving all of these billion dollar franchises. However, the caveat is that playoffs games only involve the top teams, and the games are in general much closer and competitive in terms of statistics. It would be interesting to see how accurate our proposed machine learning algorithms are then.

Finally, there is also work to be done on the comparison between machine learning predictions and Las Vegas betting. The highly simplified binary-classification model can be changed to incorporate run spreads and money lines, perhaps giving a more accurate picture of how statistics stand up to human intuition. Instead of predicting only a "win" or "loss," the algorithms must be fine-tuned to predict a better run differential or the probability that one would win a given bet. In the future, an advantageous model against Las Vegas sports books may be uncovered.

XII. Conclusion

We spent a great deal of time organizing our data and carefully choosing our features. However, we ultimately were only able to predict under 60% of games correctly. Compared to our heuristic benchmarks, we conclude that baseball games are very noisy and difficult to predict based on statistics alone. However, we were able to increase this percentage to upwards of 65% near the later portion of the season, indicating that there is some predictability in the games. For earlier games, though, it is difficult to determine outcomes from so little data. By using the simplified-classification model for betting, we also see that there is potential for using machine learning and pure statistics to advantageously bet on baseball games. With a little more adjustment, our machine learning models may obtain great financial significance.

One last comparison we can make with our work is the 2011 film *Moneyball* [4], which details the statistical analysis that went into building the 2002 Oakland A's roster. Curiously enough, the two statistics focused on by the team management – OBP and SLG – actually turned out to be rather weak features in our feature analysis. Note, though, that while our goal was to predict team wins regardless of player salaries, the goal of these statisticians was to find the most salary efficient players. So, perhaps they focused exactly on statistics that were known to be not as highly valued. That same year, the Oakland A's made the playoffs, showing that there is indeed merit in using machine learning approaches to characterize America's favorite pastime.

References

[1] Shactman, Brian A. "Big Moneyball for Major League Baseball." *CNBC.com*. N.p., 2 Oct. 2012. Web. 13 Dec. 2013. <http://www.cnbc.com/id/49259826>.

[2] "MLB Odds - Live MLB Odds." *SBRodds.com*. SBRforum, 2012. Web. 13 Dec. 2013. <http://www.sbrforum.com/betting-odds/mlb-baseball/>.

[3] "Retrosheet." *Retrosheet*. N.p., 2013. Web. 13 Dec. 2013. <http://www.retrosheet.org/>

[4] *Moneyball*. Dir. Bennett Miller. Perf. Jonah Hill and Brad Pitt. Universal, 2011.