Predicting Team Performance Based on Player History in Baseball

Nic Fishman **Rui Tang** **Rui Liu**
Department of Computer Science
George Washington University
njwfish@gwu.com rui@gwu.edu ruiliu@gwu.edu

Abstract

This paper shows the validity of using player performance histories to predict game outcomes in baseball. We show that player histories (defined as the average performance of the teams a given player has played on, in every game they've played in) are more accurate than betting markets, and at least as accurate as classification algorithms used only with team statistics as features.

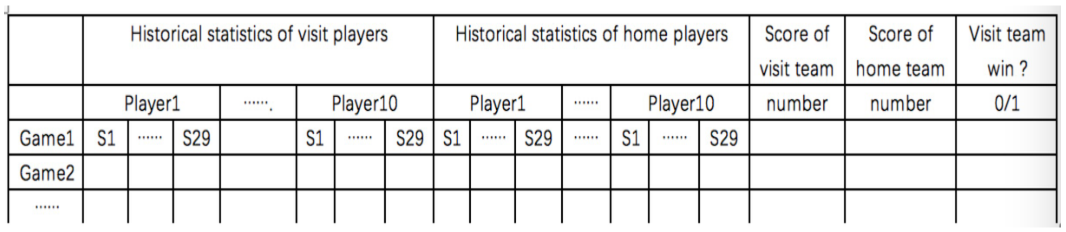
1 Introduction

Baseball is one of most popular sport in United States. According to valued at nearly 36 billion dollars[1], Major League Baseball (MLB, a a professional baseball organization) is not only an entertainment, but also an economic juggernaut. Baseball is also one of, if not the most, data rich sport. Making predictions about games' outcomes, insights about player quality, strategies for improvement are highly prized.

Betting and baseball go hand in hand, today more than ever, especially with the rise of fantasy sports, leypeople are becoming more and more involved in baseball betting, to the point it has become a multi-million dollar industry in their own right, making game prediction, especially game prediction based on player performance extremely valuable. Being able to take a set of players, and predict how they'll perform as a team is invaluable in this arena. That is the problem we're trying to solve.

There is a fairly comprehensive literature on game prediction. Finding the optimal pinch hitters, in 2003, Hirotsu and Wright did prediction based on characteristics of teams in association football by using a Markov process model[2]. Some others have attempted to predict baseball's division winners by using two stage naïve Bayes model.[3] Other game prediction research is also relevant to this problem, for example, soccer or football. In a predictive study of Finland's soccer championships, Rotshtein et al. compared the forecasting ability of both genetic algorithms and neural networks [4]. In recent years, more researches on the simulation of result were conducted. For example, Lock & Nettleton tried to use random forest algorithm to estimate win probability before each play of an NFL game in year 2014 [5].

Our research necessitated the use of a significantly more complex feature set than previous works to do the prediction. Our goal was to be able to take any set of players, from any point in their careers, and predict how they would play as a team. To that end, we gathered the data from RetroSheet, which is an online encyclopedia and data store of baseball history.[6] If we could simply train on this entire data set, we would have more than enough data points to handle our 580 features. Unfortunately, baseball is a game that has undergone profound changes since its inception. So volatile is the way the sport is played. As a result, we decided to divide the data into many batches. This issue necessitates the first step in our project plan: feature reduction. We used several techniques to reduce the dimensional of our feature vector. After this step, we had several data sets (according to the number of techniques we use for feature reduction) to be used in actual prediction. On top of that, we attempted both binary classification and regression on the data. Again, we used several techniques in



	Historical statistics of visit players							Historical statistics of home players							Score of visit team	Score of home team	Visit team win ?
	Player1			…….	Player10			Player1			……	Player10			number	number	0/1
Game1	S1	……	S29		S1	……	S29	S1	……	S29	……	S1	……	S29			
Game2																	
……																	

Figure 1: The structure of a matrix of features, after being processed.

our attempt to get the best results for our data. For this branching investigation to be meaningful, we gave each algorithm the best chance to work. In practice, that involved an in depth parameter tuning of each algorithm, which was largely omitted here, as it is fairly comprehensively covered in other literature.

Qualitatively, we show that our model can be used to predict the result of baseball games better than expert prediction and within 1% [20] accuracy of machine learning techniques to predict baseball outcomes. Further more, we compared different techniques' performance.

2 Methodology

2.1 Data Processing

2.1.1 Data Generation

Retrosheet is a volunteer run website which provides baseball statistics on all games dating back to 1871, the start of baseball. For each game, there are 29 statistics per team that have been tracked rigorously since 1912, ranging from the score, to the number of home-runs scored in a game. These logs also include the 9 starting players and the starting pitcher on each team. This gives 177645 raw games that can be used for classification.

For each game, the history of each player is taken into account to build the true feature vector to be used for classification. The average of each team statistic (score, number of home-runs, number of team errors) is taken for all games the player has played in. Because there are 10 players per team and 29 averaged features per player, there are 580 total features.

Additionally, we isolate the scores of the home and visiting teams for use in regression, and whether the visiting team won, for use in the binary classification task.

2.1.2 Batch Partitioning

Baseball changes profoundly over the course of a decade or so. New strategies emerge, rules change, the game is often very different. Longer timescales exaggerate this effect, games 100 years ago barely resemble contemporary baseball.

This means it simply doesn't make sense to run our whole data set through the various classification and regression algorithms, rather, the data must be partitioned into batches of fewer years, to mitigate the impact of changes in the game. To this end, we elected to break the data into 15 batches of 12675 games each, about 7.5 years per batch.

2.2 Feature Reduction

Summing these 29 statistics across the history of each player on each team, while comprehensive, produces more features than are useable with our small batch sizes. This requires using some sort of feature reduction technique, to ensure a proper ratio of features to data. We opted to use three algorithms: PCA, LDA, Random Projection to this end.

2.2.1 Primary Component Analysis

Primary Component Analysis (PCA) is a technique which attempts to reduce the number of features in the dataset with a minimal loss of information.[8] To achieve this, PCA involves finding the eigenvectors and eigenvalues of the covariance matrix of the whole dataset. The data projections with the highest eigenvalues are the most informative about the distribution. Additionally, this primary eigenvector can be viewed as the line along which there is the least deviation. The next eigenvector represents the direction in which those deviations are the largest in magnitude, and so on. [7]

2.2.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a model which also attempts to take linear combinations of the variables, working best when the variables are independent and continuous quantities (this is not the case in our experiment).[9][10]

In contrast to PCA, LDA attempts to preserve the information most relevant to class separation. Instead of simply choosing component axes that maximize the variance of the data, LDA also attempts to select the axes that maximize the separation between two classes.

2.2.3 Random Projection Trees

Random Projection Trees is used to reduces the dimensionality by projecting the original input space on a randomly generated matrix. [11]

2.3 Regression

Having a fairly accurate picture of how two teams will score in a given game would be very useful to the industry, consequently, we ran several regression algorithms to find the best performing one.We use the Mean Square Error (MSE) to show the accuracy of estimation.

2.3.1 Linear Regression and LASSO Regression

As what we do in the Binary classification. We use linear regression, logistic regression and LASSO regression to predict the result.

We compute the MSE of linear regression and the LASSO regression. Then we compute the classification error of both algorithms.

2.3.2 Logistic Regression

Because logistic regression can only be used to deal with the labels consisted only of 0 and 1, we decided not to divide the result into 0 and 1. Because each game goals are among the area of 0 to 14, manually make a divergence means nothing. So we only use the win or not of each game to compute the classification error of logistic regression, or the MSE of logistic can be different from others regression methods.

2.4 Binary classification

The most consequential result of these varied experiments will be what combination of classifier and dimensionality reduction can best predict which team will win. To this end, binary classification is a logical implementation, beyond just regression. The binary established here asks: did the visiting team win, with 1 indicating a yes and 0 a no, according to convention.

2.4.1 Support Vector Machines

Support Vector Machines (SVM) select a hyperplane to separate two classes to maximize the margin (the distance between the separator and the nearest point). Generally, the larger the margin, the lower the generalization error, hence its usefulness in application here.

Standard SVM constructs a linear classifier, but using the kernel trick, SVM can be used to construct more complex boundaries. The three most popular kernels are linear, Radial Basis function, and polynomial, consequently, it is these three kernels implemented here.

The main constraints to tune in SVM are the C value and whether and by what factor the kernel is scaled. The former improves generalization error, whilst the latter prevents the relative size of features from interfering with classification.

Table 1: Average average error by feature selection method. Binary Error is the average error rate of the average error rate across all batches for all binary classifiers. Regression error is the MSE on predicting team score, averaged across all batches and all regression algorithms.

	LDA	PCA	RP-Trees
Binary Error (Error rate)	0.4311	0.4404	0.4441
Regression Error (MSE)	29.7446	52.4512	9.95725

2.4.2 Naïve Bayes

Naïve Bayes is a purely probabilistic classifier, with the Naïve assumption (also known as the idiot assumption: all features are conditionally independent with respect to the classifier).

A modified version of Naïve Bayes, where instead of using a Gaussian distribution to model the continuous variables which compose the feature vector, a Kernel Density Estimation is used.[15]

2.4.3 Feedforward Neural Network

Feedforward Neural Networks (FNN) can be used for classification. Used here is a 2 layer network, with a sigmoid transfer function in the hidden layer, and a softmax transfer function in the output layer.

The number of neurons is the main parameter here to be tuned. There is not a profound amount of literature on tuning the number of neurons, so a log scale (N = 10, 50, 100, 200, 1000) was used for tuning in this case.

2.4.4 Generalized Linear Model

We use linear regression, logistic regression, Bayes regression and SGD with different C value and different penalty including LASSO and ridge. The link function used made a huge difference: linear and Bayes link functions were actually worse than random predictors. There is no obvious difference between different C value and penalty.

3 Results

3.1 Feature Reduction

For the purposes of this comparison, all feature reduction algorithms were set to produce 55 features.

3.1.1 Regression

Table 1 shows that the prediction accuracy is much higher with RP-trees than either other algorithm. This is likely because LDA is designed to separate the two classes, and PCA is designed to maintain as much information as possible, whereas RP-trees are random projections of the data, more likely to contain helpful information for regression, and less concerned with maintaining classification accuracy.

3.1.2 Binary Classification

The result captured in Table 1 relatively accurately displays the general trend: LDA almost always gives the lowest binary classification error, followed by PCA, and then RP-trees. LDA giving superior performance here is logical, designed as it is to preserve class separating information. The one caveat is that for the GLM algorithms, RP-trees often got similar, if not better results, which agrees with the results of the regression.

3.1.3 Tuning

For the rest of the paper, PCA and RP-trees were used to produce a feature set of 64 features, using Bayesian model selection to estimate the optimal number of primary components for the data. LDA remains using the 55 features originally generated.[17]

Table 2: Average MSE of each batch of data by linear regression

	Home team	Visit team
LDA	30.1044	29.3848
PCA	52.9148	51.9867
RP-Trees	9.7276	10.1869

Table 3: Average MSE of each batch of data by LASSO regression

	Home team	Visit team
LDA	28.1047	27.4828
PCA	49.9342	47.6869
RP-Trees	8.7479	9.2369

3.2 Regression

3.2.1 Mean Square Error

The results of MSE of linear and LASSO regression are shown in Table 2 and Table 3. The results reveal the data reduced by PCA gets the worst performance. While the data processed by LDA can be much better, it is again outperformed by RP-Trees, which show the best performance. The MSE of RP-trees show it's predication results can be very near to the real outcome.

In regression after LDA and PCA result, the home team has worse MSE. But in regression after RP-trees, the visit team have worse MSE. This is likely insignificant noise.

It also shows that the RP-Trees are very special. In average, the LASSO regression shows a better MSE result than what linear regression have.

3.3 Binary classification

Parameter tuning had little impact on any classifier. The most significant impact of parameter tuning was on neural nets, where the number of neurons could have up to a 15% impact on prediction, but aside from that, Naïve Bayes, SVM, and all types of GLM were largely unaffected by parameter tuning.

First and foremost from the results in Table 5, it is again clear that LDA is the best feature reduction technique for this classification task, providing the lowest error, or very close to the lowest error, in every case.

There are several standout results in the binary case. First, it is clear that the logistic regression is the best accuracy classifier. Closely followed by FNN and LASSO regression to the binary case. The fact it is so accurate when converted to binary makes LASSO probably the most useful result, as it is good both in the regression case and in simply picking a winner.

It is interesting that SVM with a 3^{rd} degree polynomial kernel gets such poor performance, whilst RBF and linear SVM are among the best performing classifiers. The extremely poor performance of the generalized linear models is also a point of interest: these classifiers are performing significantly worse than random chance, when the other regression algorithms simply converted to binary are not performing as well. This likely means this family of algorithms should be retested with a wider variety of link functions.

Figure 2 gives context for the accuracy of the most consistent classifier. FNN consistently beat the contemporary Las Vegas odds markets in every batch. It additionally illustrates the lack of any pattern over time in the difficulty of classification. There seems to be no overall trend towards either easier or more difficult to classify periods.

Table 4: Average error across all batches for all binary classification algorithms.

	LDA	PCA	RP-Trees
SVM with Linear Kernel	0.4309	0.4404	0.4453
SVM with RBF Kernel	0.4325	0.442	0.4469
SVM with Polynomial Kernel	0.4756	0.4813	0.5083
Naïve Bayes	0.4338	0.44416	0.4563
FNN	0.4134	0.4319	0.4373
Logistic Regression	0.4096	0.4876	0.4282
Linear GLM	0.5398	0.5394	0.5398
Bayesian GLM	0.5400	0.5397	0.5400
SGD GLM	0.4833	0.4798	0.4787
Linear Regression to Binary	0.4382	0.4459	0.4495
LASSO Regression to Binary	0.4282	0.4363	0.4203

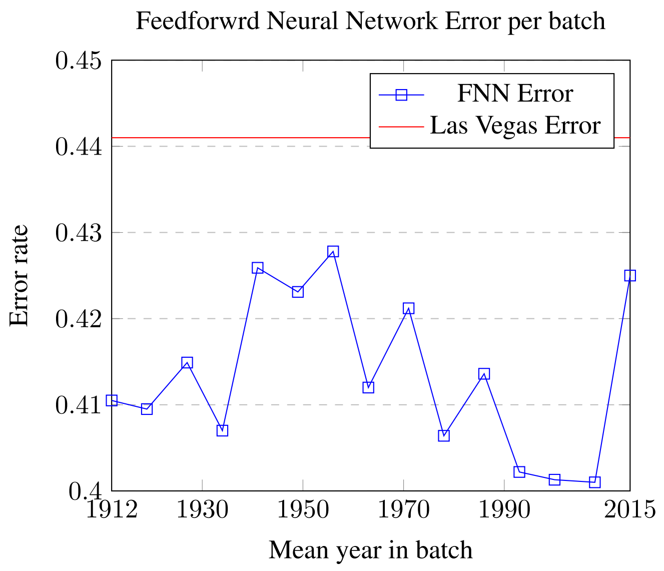


Figure 2: The FNN error on the LDA reduced data per batch. The red line indicates the average contemporary Las Vegas error in individual baseball games for comparison (calculated over the last 2 years). [18]

4 Conclusion

Here we have shown that it is possible to use a players performance history to predict, with within 1%[20] the same accuracy as using team statistics, both which team will win, and what the score spread might plausibly be. Aside from the immediate usefulness of this result, it shows the validity of our approach, and will allowing the somewhat accurate prediction of how purely theoretical sets of players might perform, as is the scenario in fantasy sports. This is the chief advantage of our approach, allowing the use of "teams" or sets of players who might never have necessarily played before.

When deploying this approach in the future, there are several modifications to be made. It is really only necessary to calculate RP-trees and Rarely does parameter tuning have much impact at all in this dataset.

Our results match the prediction accuracy of other researchers who used a more standard dataset. Although this on its own is insufficient to prove the validity of our approach, it goes a long way toward showing

Future Work
The most obvious next step is to attempt to evaluate the strength of fantasy teams using this algorithm, as that is its chief value to the industry.

We would like to look at what games the algorithm gets right and wrong compared to the betting market, to understand if it is predicting outlier games, or close games or sure things. This will determine the algorithms market value to a significant extent.

It would also be interesting to attempt extending to a different game to attempt to confirm the result that using player team performance histories is close to as accurate as simply using actual team statistics. We should make estimate the whole season performance but not as an individual game's performance.

And, we will use recurrent neural network (RNN)[19] to train our dataset, in order to get a better regression result.

Acknowledgments
We are grateful to the professor Claire Monteleoni at George Washington University for instructing us and guiding the course of this project. We are also grateful to the volunteers at Retrosheet for the compiling the comprehensive baseball dataset we used here.

References

[1] MLB Worth $36 Billion As Team Values Hit Record $1.2 Billion Average. Forbes. MAR 25, 2015

[2] Hirotsu, N. & M. Wright 2003. A Markov Chain Approach to Optimal Pinch Hitting Strategies in a Designated Hitter Rule Baseball Game. Journal of Operations Research 46(3): 353–371.

[3] Smith, L. & B. Lipscomb, et al. 2007. Data Mining in Sports: Predicting Cy Young Award Winners. Journal of Computing Sciences in Colleges 22(4): 115–121.

[4] Rotshtein, A. & M. Posner, et al. 2005. Football Predictions Based on a Fuzzy Model with Genetic and Neural Tuning. Cybernetics and Systems Analysis 41(4): 619–630.

[5] Lock, D., & Nettleton, D. (2014). Using random forests to estimate win probability before each play of an NFL game. Journal of Quantitative Analysis in Sports, 10(2), 9.

[6] Sean Smith (2009). "Retrosheet." Baseball-Reference.com - Major League Baseball Statistics and History. Retrosheet, Baseball Projection Company. <http://www.baseball-reference.com/>.

[7] Bengio, Y.; et al. (2013)."Representation Learning: A Review and New Perspectives" (PDF). Pattern Analysis and Machine Intelligence. 35 (8): 1798–1828. doi:10.1109/TPAMI.2013.50

[8] I. Jolliffe (1986). Principal Component Analysis. Springer-Verlag, New York, 1986.

[9] Abdi, H. (2007) "Discriminant correspondence analysis." In: N.J. Salkind (Ed.): Encyclopedia of Measurement and Statistic. Thousand Oaks (CA): Sage. pp. 270–275.

[10] Fisher, R. A. (1936), the Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics, 7: 179–188. doi:10.1111/j.1469-1809.1936.tb02137.x

[11] Ella, Bingham; Heikki, Mannila (2001). "Random projection in dimensionality reduction: Applications to image and text data". KDD-2001: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: Association for Computing Machinery. pp. 245–250. doi:10.1145/502512.502546.

[12] Goller, C.; Küchler, A (1996). "Learning task-dependent distributed representations by backpropagation through structure". Neural Networks, IEEE. doi:10.1109/ICNN.1996.548916.

[13] Cortes, C.; Vapnik, V. (1995). "Support-vector networks". Machine Learning. 20 (3): 273–297. doi:10.1007/BF00994018.

[14] Broomhead, D. S.; Lowe, David (1988). Radial basis functions, multi-variable functional interpolation and adaptive networks (Technical report). RSRE. 4148.

[15] John, George. H.; Langley, Pat (1995). "Estimating continuous distributions in Bayesian classifiers." Proceedings of the Eleventh conference on Uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., 1995.

[16] Nelder, John; Wedderburn, Robert (1972). "Generalized Linear Models". Journal of the Royal Statistical Society. Series A (General). Blackwell Publishing. 135 (3): 370–384. doi:10.2307/2344614. JSTOR 2344614.

[17] Minka, Thomas P. "Automatic choice of dimensionality for PCA." In NIPS, vol. 13, pp. 598-604. 2000.

[18] “MLB Odds - Live MLB Odds.” SBRodds.com. SBRforum, 2012. Web. 13 Dec. 2013. <http://www.sbrforum.com/bettingodds/mlb-baseball/>.

[19] Goller, C.; Küchler, A (1996). "Learning task-dependent distributed representations by backpropagation through structure". Neural Networks, IEEE. doi:10.1109/ICNN.1996.548916.

[20] Jia, R.; Wong, C.; Zeng, D. (2012). "Predicting the Major League Baseball Season" Stanford CS 229.