Predicting Pitchers' Early Career Value From Rookie Year Performance

Joey Asperger and Austin Poore
{joey2017, hapoore}@stanford.edu

Predicting

Major League Baseball is big business, to the tune of billions of dollars per year in revenue, and baseball, as a game that can effectively be broken down into a series of individual actions, rather than more complicated interactions between players that govern sports like football, basketball, and soccer, is uniquely positioned for advanced statistical analysis. One interesting metric to gauge player value is called Wins Above Replacement (WAR), which, intuitively, seeks to answer the question of how many more games my team would lose if I were replaced by a readily-available player from my team's bench or the minor leagues. Initially, our plan was to use features derived from pitchers' performance from their rookie seasons to predict their WAR for years 2-4 of their careers, but that distribution is very right-skewed, and we realized that our models were having trouble dealing with a relatively small number of outliers, which were being penalized heavily. We also realized that we care less about whether a player is worth 3 wins or 4 wins over a three-year period (a decent, but not spectacular player), and more about whether or not he's likely to be an excellent player during that period. Thus, we switched our focus from building a regression model that could predict WAR effectively to a classifier that sought to identify these future star players.

Data

Our features dataset, taken from Sean Lahman's database, came in the form of a table where each row contained a pitcher's statistics for a given year, including stats like wins, losses, ERA, and other traditional pitching statistics. For our labels, we used a separate source, Baseball Reference, to obtain each player's yearly WAR. We then labeled players as high-value if their total war over the next 3 years was greater than 5, and low-value if it was less. Our total training dataset contained 813 players, 193 of whom were high-value, and our test set had 204 players, 53 of whom were high-value.

Features

For our features, we selected selected important stats from each pitcher's rookie season, shown in the table below. We then tested the relevance of each feature using cross-validation. In addition, we tested models that transformed the features with gaussian or polynomial kernels as well as with principle components analysis.

Wins	Losses	ERA	Win %
Innings	Strikeouts	SO/IP	Games
Hits	Hits/inning	Home runs	HR/IP
Walks	Walks/IP	Opp BA	R
ER	WAR	Age	Team defensive runs saved

Models

Support Vector Machine

We used support vector machines as one of our models because the hinge loss function seemed like a reasonable choice in our objective function. Recall that the hinge loss penalizes based on the size of the margin.

$$\mathsf{L}(z,y) = [1-yz]_+ = \max\{0, 1-yz\}$$

SVMs also had the benefit of being able to use different kernels to generate nonlinear decision boundaries, so we tested using with linear, gaussian, and polynomial kernels. For the polynomial kernel, our model suffered from overfitting, so we also tested using principle component analysis to reduce the dimensionality of the feature vector from 20 to 12 before fitting the SVM.

Boosting

We knew this problem was challenging, so we were also interesting in trying out boosting to see if some combination of weak learners would be able to perform well. Recall that we seek to minimize the following objective, where ϕ represents our set of weak learners.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \exp(-y^{(i)}\theta^T\phi(x^{(i)}))$$

We ran our boosting model with 100 weak learners, so as to try to avoid overfitting our data.

Results

Model	Training Error	Test Error
SVM - Linear	29.6%	29.9%
SVM - Polynomial	12.4%	32.3%
SVM - Gaussian	27.4%	29.9%
SVM w/ PCA - Polynomial	21.7%	29.4%
Boosting	8.2%	24.5%
Boosting - oversampling	10.9%	34.3%
Boosting - undersampling	18.6%	32.4%

The column on the right contains confusion matrices for three of our classifiers on the test set. We found it particularly informative to see the difference in undersampled boosting, and to see that accuracy doesn't tell the whole story.

Predicted

Actual	High	Low
High	30	23
Low	38	113

SVM-Gaussian

Predicted

Actual	High	Low
High	11	42
Low	8	143

Boosting

Predicted

Actual	High	Low
High	30	23
Low	43	108

Boosting - undersampling

Discussion

Our performance is obviously not great, and our biggest takeaway from this project is that predicting WAR is a challenging problem, and there's probably a reason Major League front offices employ teams of data scientists to work on problems like this in search of a competitive edge. Our best models got to around 25-30% error, and we were not particularly surprised at this performance given the many things that could go wrong over the first few years of a pitcher's career. For instance, a nagging injury during a pitcher's rookie year might cause him to look very unimpressive but then he could turn into an All-Star after dealing with the issue, or a player could miss most of a year due to an injury or be moved to the bullpen by his manager, both of which would likely affect his WAR but be unpredictable with metrics like those we used for our features. One challenge we faced was the fact that the majority of players are not great, so our data had many more examples of average players than stars. We tried oversampling and undersampling for our boosting classifier, which made the misses more uniform (without it, we simply tended to way overpredict the larger class) but decreased overall accuracy, and weighted the examples for our SVM such that both classes carried equal weight, which improved our performance. We used a library called Optunity to help tune our SVM hyperparameters using cross-validation on our training set. For our boosting classifier, we also used cross-validation on our training set to decide on the number of weak learners to employ, and settled on the SKLearn default of 100 in order to avoid potential overfitting.

Future Work

Given several more months to work on the project, we would concentrate initially on feature selection. As mentioned in the discussion portion, it's tough to predict performance over a period of several years because so many random things can happen, like injuries, so we suspect that features that can predict things like that to some degree would be helpful. We'd also like to run some experiments to figure out which of our current features are actually informative, and if there are any we can get rid of. Additionally, WAR calculations depend on how good your team's defense is, where you are playing, and the quality of the opposing hitters you've faced, so we'd like to incorporate more features that attempt to capture more of those components, or perhaps experiment with other metrics of performance in addition to WAR.

References

Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

Sports Reference LLC. Baseball-Reference.com - Major League Statistics and Information. http://www.baseball-reference.com/. 11 Dec. 2016.

Lahman, Sean. "Download Lahman's Baseball Database." *Sean Lahman | Database Journalist.* Web. http://www.seanlahman.com/baseball-archive/statistics/. 11 Dec. 2016..