Austin Poore
SUID: hapoore
05865600

Joey Asperger
SUID: joey2017
05859436

Predicting Pitchers' Early-Career Value From Rookie Year Performance

Abstract:

We seek to develop accurate models for predicting the early-career value of pitchers in Major League Baseball (MLB) based on statistics and data from their first season. More specifically, we use Wins Above Replacement (WAR) as a measure of player value, and our goal is to be able to accurately predict whether or not a pitcher will be worth at least five wins (cumulative WAR > 5) in years 2-4 of his career, based on features from his first year. We apply a variety of feature selection and evaluation techniques on a variety of different classification models in order to determine the best way to make these predictions. This problem proved to be surprisingly challenging using feature vectors consisting mostly of traditional pitching statistics, which provides motivation for future work in deriving more informative metrics that generalize better to future success.

Introduction

Major league baseball players typically become free agents for the first time after their first few years in the MLB, so our goal is to make predictions about a pitcher's cumulative Wins Above Replacement (WAR) during the period where he is still under the control of his first team, based on data and statistics from his rookie year. More specifically, we'd like to make predictions about a pitcher's value over the second, third, and fourth years of his career. WAR is a statistic intended to be an all-inclusive measure of a baseball player's value; intuitively, it seeks to answer the question "How many more games does a player's team win by having that player rather than a readily-available minor league replacement?" A player with a WAR of 2.0, for instance, is thought to be worth two more wins to his team over the course of a season than a readily-available replacement player. Thus, it provides a decent estimate of a player's true value (much more so than some other traditional metrics). The formula to compute WAR is complex, and not particularly valuable to look at in full for our purposes, but it is important to note that it depends on a pitcher's statistics, as well as the quality of opponents he's faced, the quality of his team's defense, and where he played, mostly in order to be able to make comparisons to the hypothetical league-average pitcher.

Initially, we tried to train regression models to make predictions about players' WAR during this early-career period. However, we realized that our data set was extremely right-skewed; most players ended up with very similar WAR values, with only a handful of standouts far exceeding the rest by a wide margin. We tried several different regression models, but all had trouble dealing with these few outliers, and we decided that perhaps being able to predict a one or two-win difference over a three-year period was less useful than being able to identify future superstars. Thus, we switched our focus to distinguishing between two classes of players: the first consists of average to decent players, who might be solid Major League contributors but are not superstars, whereas the second consists of the future All Stars, Cy Young winners, and MVPs. We defined any player worth at least five wins over the three-year period as a member of our second class, which was right above the third quartile of our data set. The inputs to our classifiers were feature vectors

made up mostly of aggregated statistics from players' rookie seasons. The classifiers would then output a label, either zero or one, where a one represented a predicted future star.

Review of related literature

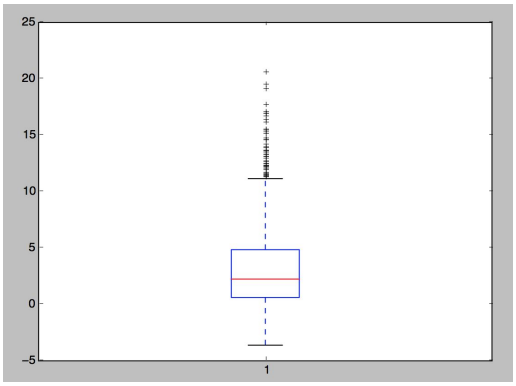
After doing some digging, we struggled to find much existing research that attempted to make the predictions we hoped to be able to make. In some ways, this is not entirely surprising; it is almost certain that all (or nearly all) MLB teams are employing analysts internally to apply statistical methods to evaluate players. As these discoveries represent a competitive edge, however, results are unlikely to be shared. Baseball fans are notorious statheads, though, so there are many blog posts and senior theses out there about people attempting to predict player performance in some sense, like [1], [2], and [3].

However, our premise is a bit different in a couple of important ways. First of all, predicting performance for pitchers is more difficult than for position players. As noted in [1], there's an old saying that "There's no such thing as a pitching prospect," because a pitcher can blow out his arm on any single pitch, and there are a lot more injuries common to pitchers that can completely derail a career than for position players. Second, much of the existing analysis we turned up sought to predict a player's MLB value or performance (or even whether they were likely to make the major leagues) based on minor league statistics, whereas we hope to use early MLB statistics to predict future value. These are inherently different problems, though some techniques might be useful to both.

One problem we faced during our project was the fact that we had many more examples of regular players as opposed to future stars, which meant that initially, our classifiers simply learned to predict mostly the larger class. In order to combat this for our boosting algorithm, we employed some of the undersampling and oversampling techniques described in [4], which tended to hurt our accuracy but improve our recall. We also weighted training examples differently when training our SVMs, attaching more weight to examples from the smaller class.

Dataset and Features

Our features came primarily from Sean Lahman's database, a free online database that compiles MLB stats dating back to 1871. This database contained pitching statistics for each year, including stats like wins, losses, ERA, and other traditional metrics. For our labels, we used a separate source, Baseball Reference, to obtain each player's yearly WAR. We then labeled rookies as high-value if their total war over the next 3 years was greater than 5, and low-value if it was less. Originally, we had hoped to also use data from the PITCHF/x dataset, which contains very advanced information on every pitch throw in in an MLB game, but since this system is relatively new, we would only have 5 years of training data to work with, which we did not think would be enough to adequately train our classifiers. Ultimately, we decided to use data going back to 1985, which would give us a large training set, but not include data that is so old it wouldn't be relevant to modern pitchers. Our overall data set consisted of 1,017 pitchers total (all of the rookies from 1985-2013 who played in each of the next three years), 813 of which we used as training examples (193 high-value) and 204 of which we used as test examples (53 high-value). See the attached box plot of our data set:



For our features, we selected selected 20 important stats from each pitcher's rookie season, shown in the table below. Our goal in selecting features was to make sure our feature set was informative enough to make predictions but not too large that it suffered from overfitting. For each feature, we also normalized the data points to have a mean of zero and a standard deviation of one. We also transformed the feature vector in some of our models, using Gaussian and polynomial kernels for our SVM. In addition, we also used principal components analysis in our polynomial kernel model to reduce the dimensionality of our feature vector from 20 to 12 in order to avoid overfitting.

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 runs saved

Methods

We used the scikit-learn machine learning library for Python, cited in [5], to build and test all of our models.

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:

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

Our intuition was that a larger margin means that we're more confident in our predicted classification, so optimizing for large margins seems like a reasonable choice. SVMs also have the benefit of being able to use different kernels, like the Gaussian kernel, or a polynomial kernel, to generate nonlinear decision boundaries, which would help us capture patterns which we would not be able to with a simple linear decision boundary. In our model, we also increased the weight for our data points of high-impact players in order to balance to two classes. In order to find the optimal hyperparameters for the model, we used a library called Optunity and tested the hyperparameters using 5-fold cross-validation and scoring models based on the area under the curve plotting true positives against false positives.

We also used principal component analysis with some of our SVM models, which reduces the dimensionality of our training data by projecting the feature vectors onto a lower dimensional subspace that still captures most of the variance in the data. Recall that in PCA, we find the top k eigenvectors of the matrix Σ, where $\Sigma = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}x^{(i)T}$, and project our data onto the subspace spanned by these k vectors. As an example, the top eigenvector is the unit-length vector u that maximizes the following equation:

$$\frac{1}{m}\sum_{i=1}^{m}(x^{(i)^T}u)^2$$

The next PCA vector maximizes the same equation given the extra constraint that it be orthogonal to the first, and so on for the top k.

We also used boosting as one of our models because for such a challenging problem, it might be easier to make predictions using a combination of weak learners. Boosting works by combining many weak learners that do slightly better than random, giving more weight to those that do well on examples in the training set we are getting wrong

and resulting in an overall classifier that is able to do much better than any individual weak learner. 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, which is a concern with boosting, because given enough learners the algorithm is able to learn the training set perfectly, but then will likely fail to generalize to unseen test data.

We ran boosting with both oversampling and undersampling techniques, as well as with the unmodified training set. In order to oversample, we replicated the training examples from star players and added these duplicates to the training set until we had a roughly even split between the two classes, whereas when we undersampled we threw away non-superstar examples until we had an even split.

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%

As can be seen in the table above, our linear and gaussian SVMs performed reasonably well, getting just under 30% error. For the the SVM with a polynomial kernel, it suffered significantly from overfitting, getting 12.4% training error and 32.3% test error. In order to fix this problem, we reduced the dimensionality of the feature vector using PCA, which significantly improved the model. Below, we show a confusion matrix of our Gaussian SVM model.

Predicted

Actual		High	Low
	High	30	23
	Low	38	113

SVM-Gaussian

As you can see, our model was fairly balanced between precision and recall.

For our boosting model, it initially got better test error than any of our other models. However, we realized that this was a result of the model labeling almost all of the data points as the majority class, which is not actually useful. Its confusion matrix is shown below

Predicted

Actual		High	Low
	High	11	42
	Low	8	143

Boosting

In order to solve this issue, we tested both oversampling and undersampling, which had slightly worse test error, but better (more balanced) confusion matrices. The confusion matrix for undersampling is shown below.

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.

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. During the poster session, Professor Duchi also pointed out that features with some sort of temporal aspect might also be useful in making predictions, in order to identify trends in a player's performance throughout his rookie year. For instance, if a player started out poorly but then came on strong the second half of the year, that might be more informative than seeing all of his stats aggregated for the entire year. Data like this is much less readily available, and would be quite time-consuming to scrape, but it would be interesting to explore nonetheless.

References

[1] Mitchell, Chris. “Forecasting Major League Hitting with Minor League Stats”. *The Hardball Times.* Web. Dec. 30 2014. <http://www.hardballtimes.com/katoh-forecasting-a-hitters-major-league-performance-with-minor-league-stats/>

[2] Pinheiro, Ryan Xavier. “Efficient Free Agent Spending in Major League Baseball”. (Master’s thesis) Web. <https://etd.ohiolink.edu/!etd.send_file?accession=akron1396821766&disposition=inline>

[3] Tymkovich, Jay Lyon, "A Study of Minor League Baseball Prospects and Their Expected Future Value" (2012). *CMC Senior Theses.* Paper 442. <http://scholarship.claremont.edu/cmc_theses/442>

[4] C. Drummond, R.C. Holte, “C4.5 Class Imbalance and Cost Sensitivity: Why Under Sampling Beats Over-Sampling”. *Proc. Int’l Conf. Machine Learning Workshop Learning from Imbalanced Data Sets II*. 2003.

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

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

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