Predicting the Next Pitch

Gartheeban Ganeshapillai, John Guttag
Massachusetts Institute of Technology,
Cambridge, MA, USA, 02139
Email: garthee@mit.edu

Abstract

If a batter can correctly anticipate the next pitch type, he is in a better position to attack it. That is why batteries worry about having signs stolen or becoming too predictable in their pitch selection. In this paper, we present a machine-learning based predictor of the next pitch type. This predictor incorporates information that is available to a batter such as the count, the current game state, the pitcher's tendency to throw a particular type of pitch, etc. We use a linear support vector machine with soft-margin to build a separate predictor for each pitcher, and use the weights of the linear classifier to interpret the importance of each feature.

We evaluated our method using the STATS Inc. pitch dataset, which contains a record of each pitch thrown in both the regular and post seasons. Our classifiers predict the next pitch more accurately than a naïve classifier that always predicts the pitch most commonly thrown by that pitcher. When our classifiers were trained on data from 2008 and tested on data from 2009, they provided a mean improvement on predicting fastballs of 18% and a maximum improvement of 311%. The most useful features in predicting the next pitch were Pitcher/Batter prior, Pitcher/Count prior, the previous pitch, and the score of the game.

1 Introduction

Batters often "sit on a pitch." That is to say, they guess that a pitch will have certain attributes, e.g., be a fastball, and prepare themselves to swing if their guess is correct and not swing otherwise [1][2].

In this paper, we present a pitcher-specific machine-learning based system for predicting whether the next pitch will be of a specific type (e.g., a fastball). This predictor incorporates some information about the current at bat and game situation (e.g., the previous pitch, the count, the current score differential, and the current base runners) [3] and some information about the pitcher's tendency to throw a pitch (prior) with that property in various situations.

We evaluated our method on an MLB STATS Inc. dataset, which contains a record of each pitch thrown in both the regular and post seasons [4]. We trained our model using the data from 2008, and tested it on the data from 2009. For predicting whether the next pitch would be a fastball, our classifier predicts, on average, 70% of the pitches accurately. This represents a mean improvement of 18% over a naïve model that uses the pitcher's historical fastball frequency to predict the next pitch

type. For the 187 pitchers whose historical fastball pitch frequency was between 30% and 70%, the mean improvement is 21%. For the 100 pitchers for whom the algorithm performed the best, the improvement averaged 43.5%.

2 Method

We pose the prediction of the next pitch as a binary classification problem. For instance, we try to predict whether the next pitch is a fastball or not, and use a binary classifier to model it. In Section 3, we present the results for six different pitch types. However, for simplicity of explanation, we will talk only about predicting fastballs in this section.

Table 1 lists the features in the feature vector that forms the independent variable used to predict the dependent variable, i.e., the next pitch.

Table 1. Feature Vector

Game performance	Balls	Strikes	Outs	Score differential	Bases loaded
Game state	Inning	Handedness	Number of Pitches thrown		
Prior probability	Home team	Batting team	Count	Batter	Defense formation
Support of the priors	Home team	Batting team	Count	Batter	Defense formation
Batter Profile	Slugging Percentage	Runs	For each pitch class		
			Runs	Slugging percentage	
Previous Pitch	Pitch Type	Pitch Result	Velocity	Vertical Zone	Horizontal Zone
Gradient over last 3 pitches	Velocity	Vertical Zone	Horizontal Zone		

Handedness denotes whether the pitcher and batter are of the same handedness. Prior probabilities are computed for all pitcher/variable pairs (e.g., pitcher/home team and pitcher/batter pairs). Score differential is the absolute value of the difference between the runs scored by both teams.

We build a separate model for each pitcher and train it using historical data. For each pitch thrown by that pitcher we derive a feature vector and associate a binary label (true for fastball and false for non-fastball) with that sample.

The samples described by these feature vectors are not perfectly linearly separable. That is to say there does not exist a hyperplane such that all of the samples with a positive label lie on one side of the hyperplane and all of the samples with a negative label lie on the other. At first blush, this suggests that one should use a non-linear classifier, e.g., a support vector machine with a Gaussian kernel. Unfortunately, however, such classifiers produce a model that is difficult to interpret, i.e., it is hard to tell which features contribute most to the result. Consequently, we use a linear support vector machine classifier with soft-margin [5] to build our models.

A support vector machine is a classifier that, given a set of training samples marked as belonging to one of two categories, builds a model that assigns new samples into one category or the other. It constructs a hyperplane that separates the samples belonging to different categories. It minimizes the generalization error of the classifier by maximizing the distance to the hyperplane from the nearest

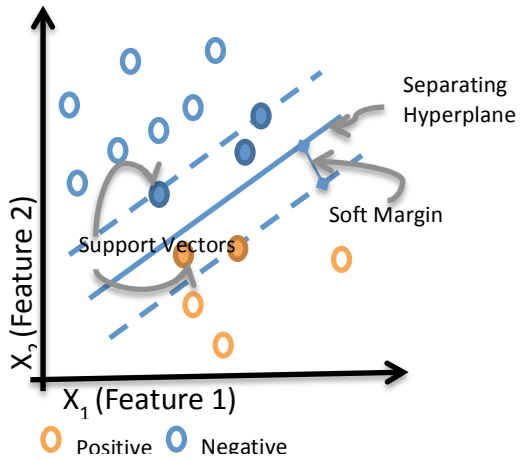

Figure 1. Support Vector Machine and soft-margin

samples on either side. The samples that define the hyperplane of the classifier are called support vectors (SV), and hence the classifier is named a support vector machine (see figure 1) [5].

When the samples are not linearly separable, a linear support vector machine with soft margin will choose a hyperplane that splits the examples as cleanly as possible, while still maximizing the distance to the nearest examples (see figure 1).

Sometimes, the values of the features such as pitcher/batter prior probability might be empty or unreliable because of small support, e.g., a particular pitcher might not have faced a particular batter, or thrown only a few pitches to that batter. In such cases, the prior probability may not be meaningful. In such situations, values with low supports can be improved by shrinkage towards the global average [7]. The global average can be obtained from the pitcher's overall prior probability. For the pitcher-variable prior s, global average p, support n, and some constant β, the shrunk pitcher-variable prior $\hat{s}$ is given by

$$\hat{s} = \frac{n\,.\,s + \beta\,.p}{n + \beta}$$

3 Results

We use SVM-Light tool to build our models [6]. We use the records from year 2008 to train our model, and the records from 2009 to test. We compare our method's accuracy (A_o) against the accuracy of a naïve model (A_n) that uses each pitcher's prior probability, i.e., the likelihood from his history. We measure the usefulness of our method by improvement, I over the pitcher's prior.

$$I = \frac{A_o - A_n}{A_n} \times 100\%$$

3.1 Predicting the next pitch type

We train a binary classifier to predict whether the next pitch is going to be a fastball or not. We consider those 359 pitchers who threw at least 300 pitches in both 2008 and 2009. The average accuracy of our model is 70%, compared to the naïve model's accuracy of 59.5%. Average improvement on a by pitcher basis was 18% across all the pitchers. On pitchers with a prior probability of throwing a fastball between 0.3 and 0.7, the average improvement is 21%, and for the 100 pitchers for whom we got the highest improvement the average improvement is 43.5%. For pitchers with a very high prior there is not much room for improvement. For example, for *Mariano Rivera* our model improved the accuracy from 92.1% (naïve model) to 94.1%. Since our model considers several factors, it performs well even when pitchers change their dominant pitch types. For instance, even when the frequency of fastballs thrown by *Andy Sonnanstine* changed from 61% in 2008

to 7% in 2009, our model achieved 32% accuracy in predicting the next pitch, where as the naive model managed only 8% accuracy.

Table 2. Performance on specific pitchers.

	Name	ERA (2009)	Pitches		Accuracy		I
			Training	Test	A_o	A_n	
Greatest Improvement	Andy Sonnanstine	6.77	3125	1675	32%	8%	311%
	Brian Bannister	4.73	3074	2475	47%	16%	196%
	Miguel Batista	4.04	2022	1169	72%	36%	100%
	Scott Feldman	4.08	2356	3145	65%	35%	85%
	Rafael Perez	7.31	1098	694	73%	40%	79%
	Francisco Rodriguez	3.71	1109	1154	71%	44%	62%
	Kyle McClellan	3.38	1128	1060	76%	49%	53%
	Nick Blackburn	4.03	2830	3154	62%	41%	52%
Highest Accuracy	Mariano Rivera	1.76	911	1202	94%	92%	1%
	Tim Wakefield	4.58	2693	1998	93%	89%	4%
	Mark DiFelice	3.66	318	742	92%	92%	0%
	Bartolo Colon	4.19	550	974	90%	90%	0%
	Roy Corcoran	6.16	1066	304	89%	88%	1%
	Matt Thornton	2.74	1013	1095	88%	88%	0%
	Aaron Cook	4.16	2996	2448	86%	86%	0%
	Jesus Colome	7.59	1131	334	84%	79%	6%
Least Accuracy	Andy Sonnanstine	6.77	3125	1675	32%	8%	311%
	Brian Bannister	4.72	3074	2475	47%	16%	196%
	Chad Durbin	4.39	1371	1290	56%	49%	16%
	Francisco Cordero	2.16	1176	1012	58%	39%	48%
	Jason Jennings	4.13	506	1025	59%	51%	14%
	Braden Looper	5.22	3168	3214	59%	48%	22%
	Darren Oliver	2.71	1052	1187	59%	47%	27%
	Cliff Lee	3.22	3235	4068	59%	59%	0%

3.2 Most useful predictors

In our pitcher-specific model, we use a linear classifier. A linear classifier's weights can be interpreted to identify the most useful predictors. Figure 2 shows the distribution of weights (mean and standard deviation) across pitchers, and Table 3 lists the top 12 predictors and their mean weights across all the pitchers.

Notice that home team does not appear in this table, suggesting that the stadium in which the game is played does not seem to have much impact on pitch selection.

Table 2. Top 12 predictors

Predictor	Weight
Pitcher – Batter prior	0.4022
Shrunk Pitcher – Batter prior	0.2480
Pitcher – Count prior	0.2389
Shrunk Pitcher – Count prior	0.2238
Previous pitch's velocity	0.1529
Velocity Gradient	0.1359
Previous pitch type	0.1138
Inning	0.0650
Outs	0.0522
Score difference	0.0408
Bases occupied	0.0398

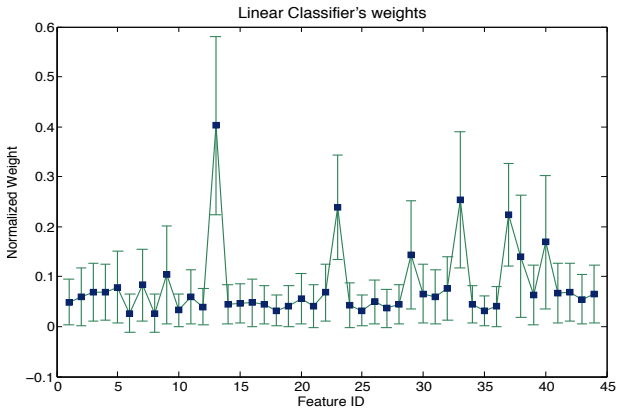

Figure 2. Distribution of classifier weights

3.4 Predictability

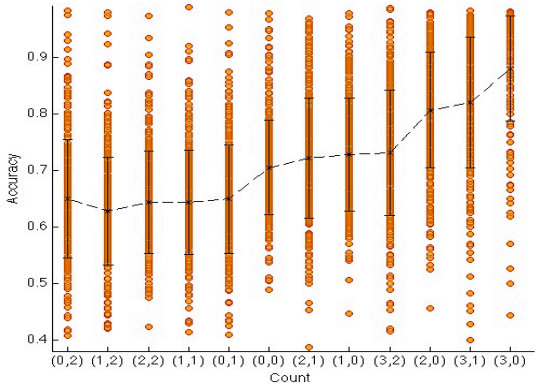

Figure 3. Predictability against count

Next, we test the pitchers' predictabilities in various situations. We use our method's accuracy to quantify the predictability of a pitcher under various circumstances.

First, we look at count (balls, strikes) ordered by its favorability towards the batter. Figure 3 is the scatter plot of the accuracy against the counts across pitchers with the mean and standard deviation plotted on top. This pattern shows that pitchers are more predictable, at less favorable counts (to the pitcher) such as (3,0) compared to more favorable counts such as (1,2).

Next, we investigate whether a high score differential makes a pitcher more or less predictable. We look at two cases: the score differential is less than three (low), and the score differential greater than three (high). We observe a pattern when we consider their variations against the inning (Figure 4). At the beginning of the game (before the 4th inning), accuracy is statistically significantly different (P-value 0.01) between these cases, i.e., the pitcher is less predictable when the score difference is high. This difference, however, disappears in the latter innings. This suggests that perhaps starting pitchers are more likely to vary their pitch selection based on the score than are relievers.

Since there is not much room for improvement on pitchers with a high prior, we can expect the improvement to be low for these pitchers. However, the improvement has a clear downward trend against the pitcher's prior, starting as early as at 0.5 (Figure 5). A plausible explanation is that the pitchers with a dominant pitch type trust that pitch, and don't change their pitch selection based on other factors. Hence, the improvement is low for these pitchers.

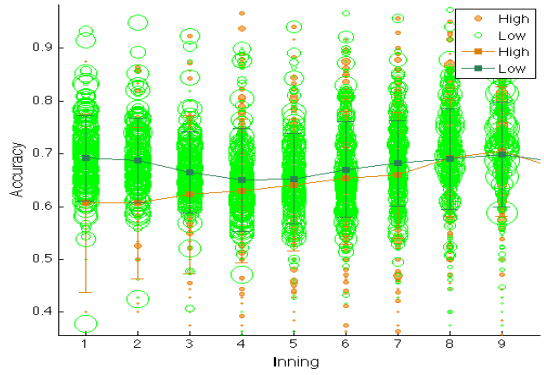

Figure 4. Influence of score differential and innings on accuracy

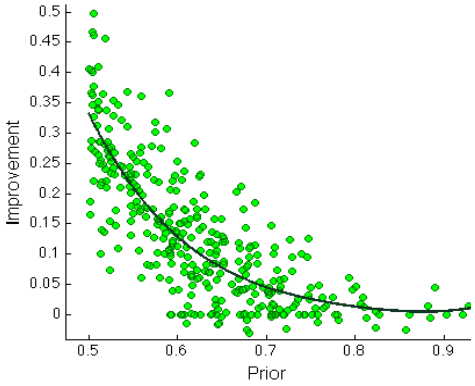

Figure 5. Relationship between prior and improvement

We didn't observe any significant patterns on the predictability against other features such as batters' OPS, home team, or defense configuration. Our model's performance is independent of pitcher's ERA. Further, the mean accuracy was independent of both whether the pitcher was a starter, closer, or other relievers and of the number of pitches per pitcher in the training set.

3.3 Other pitch types

We also tested our method for other pitch types. Table 4 compares the improvement in the accuracy achieved by our model on different pitch types. Here, n is the number of pitchers. There is no pitcher with prior probability between 0.3 and 0.7 for sinker or knuckleball.

Table 3. Comparison with other pitch types

	Prior between 0.3 and 0.7				Prior between 0.4 and 0.6	
Pitch Type	**n**	**A_o**	**A_n**	**I**	**n**	**I**
Fastball	279	68.2%	57.9%	19.5%	153	26.2%
Changeup	11	67.2%	64.8%	3.9%	1	12.6%
Slider	54	64.4%	61.5%	6.3%	16	16.2%
Curve	16	65.7%	64.3%	2.5%	3	12.1%
Split-Finger or Forkball	3	73.4%	68.0%	8%	0	-
Cut Fastball	9	56%	54.4%	3.4%	5	6.2%

4 Conclusion

While most pitchers have a dominant pitch, there are clearly other factors that influence their pitch selection. For many pitchers these factors can be used to significantly improve one's ability to predict the type of the next pitch.

We make no claim for the optimality of our choice of features. In fact, we expect that further study will lead to feature vectors that yield better performance.

Acknowledgement

We would like to thank STATS Inc. for providing us with the data. This work was supported by Quanta Computers Inc.

References

[1] Smith, David W. "Do Batters Learn During a Game?" Web. June 7, 1996. <http://www.retrosheet.com. September 8, 2010>
[2] Laurila, David. "Prospectus Q & A: Joe Mauer." Baseball Prospectus. 8 July 2007. Web. 5 Jan. 2011. <http://www.baseballprospectus.com/article.php?articleid=6428>
[3] Appleman, David. Pitch Type Linear Weights. Fangraphs. 20 May 2009. Web. 5 Jan. 2011. <http://www.fangraphs.com/blogs/index.php/pitch-type-linear-weights>
[4] STATS. Inc. Web. 5 Jan. 2011.<http://www.stats.com/baseball.asp>
[5] Mukherjee. S and Vapnik. V, "Multivariate density estimation: A support vector machine approach". Technical Report, AI Memo 1653, MIT AI Lab
[6] T. Joachims, et al, Making large-Scale SVM Learning Practical, "Advances in Kernel Methods - Support Vector Learning", MIT-Press, 1999.
[7] Robert M. Bell and Yehuda Koren, "Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights". In Proceedings of the 2007 Seventh IEEE International Conference on Data Mining (ICDM '07).